DDS5

Vector Controlled Stepper Motors Drives

with Modbus RTU

User Manual

(Hardware rev. 1.00 Firmware rev. 0.11)

1 Introduction

The DDS5 drives series is equipped with Modbus RTU and drives the stepper motors with vector technique.

The I/O equipment is complete and includes digital and analog inputs and outputs.

Some models have the encoder input, which allows the closed-loop motor control thus eliminating stall problems and improving efficiency.

The supported operative modes allow the control of the motor in position, speed and torque. Numerous homing modes are also available.

The free software *Omni Automation IDE*, running under Windows platform (Windows 7, Windows 8.1 and Windows 10 32bit or 64bit), assists the commissioning of the product. For the connection to the PC the UDP30 interface is needed.

1.1 Series

The series develops in 20 models, different in functionalities and power.

Power Supply / Motor Current 24Vdc Auxiliary Power Supply	6 Digital Inputs 2 Digital Outputs	6 Digital Inputs 3 Digital Outputs 2 Analog Input 2 Analog Output 1 Encoder Input A, B, I	6 Digital Inputs 3 Digital Outputs 2 Analog Input 2 Analog Output 1 Encoder Input A, B, I 1 Encoder Input SSI	
20..50Vdc / 0.2..1.4Arms	**DDS5041**	**DDS5241**	**DDS5441**	DC Power Supply
20..50Vdc / 1.0..4.5Arms	**DDS5044**	**DDS5244**	**DDS5444**	
20..50Vdc / 2.0..10.0Arms	**DDS5048**	**DDS5248**	**DDS5448**	
24..90Vdc / 1.0..4.5Arms	**DDS5074**	**DDS5274**	**DDS5474**	
24..90Vdc / 2.0..10.0Arms	**DDS5078**	**DDS5278**	**DDS5478**	
16..36Vac / 0.2..1.4Arms	**DDS5041A**	**DDS5241A**	**DDS5441A**	AC Power Supply
16..36Vac / 1.0..4.5Arms	**DDS5044A**	**DDS5244A**	**DDS5444A**	
16..36Vac / 2.0..10.0Arms	**DDS5048A**	**DDS5248A**	**DDS5448A**	
20..65Vac / 1.0..4.5Arms	**DDS5074A**	**DDS5274A**	**DDS5474A**	
20..65Vac / 2.0..10.0Arms	**DDS5078A**	**DDS5278A**	**DDS5478A**	

1.2 Terms, symbols and abbreviations

To indicate features common to a whole group of products the character “x” is used in place of any other character. For example, the term DDS5x44 implies the models DDS5044, DDS5244 and DDS5444.

The terms manual and document have the same meaning, moreover the words drive, device and product always refer to the DDS5 Series.

In the manual some symbols are used to underline necessary topics of particular concern or deserving interest. The meaning of each one of them is detailed here below.

It refers to a dangerous condition that must be accurately evaluated and avoided. Failing to follow instructions marked with this symbol can be cause of serious damages to people, animals and things.

It draws the attention to important issues that if not understood or implemented may affect the good functioning of the product.

It highlights a valuable feature or functionality of the product that is difficult to find elsewhere or shows a shortcut to reach a target.

The characteristic names of registers, parameters, objects, modes, etc. are always provided in English to avoid confusion or doubts in the interpretation.

To describe the data type of registers, parameters, objects, etc. abbreviations are used. The following table shows their meaning, as well as the range of values allowed for each type:

Abbreviation	Bits	Description	Min.	Max.
i8	8	Signed Byte	-128	127
i16	16	Signed Word	-32.768	32767
i32	32	Signed Integer	-2.147.483.648	2.147.483.647
u8	8	Unsigned Byte	0	255
u16	16	Unsigned Word	0	65.535
u32	32	Unsigned Integer	0	4.294.967.295
f32	32	Floating Point	-3,402823e38	3,402823e38
str	---	Stringa		

The symbols ↑ and ↓ are used to indicate respectively the rising edge (0->1 transition) or the falling edge (1->0 transition) of a digital signal.

Other abbreviations used:

Abbreviations	Description
AC, ac	Alternate current
AI	Analog input
AO	Analog output
DC, dc	Direct current
hm	Homing mode
DI	Digital input
DO	Digital output
pp	Position mode
pv	Velocity mode
RMS, rms	Root mean square
RO, ro	Read-only
WO, wo	Write-only
RPD	Receive Process Data
RW, rw	Read-write
TPD	Transmit Process Data
tq	Torque mode
ADU	Application Data Unit
MBAP	MODBUS Application Protocol
PDU	Protocol Data Unit

1.3 Documents

The present manual applies to the standard series of DDS5 drives with Hardware and Firmware revisions as shown on the cover. Customized products or with a different Hardware or Firmware revision may have features and behaviors different to what herein described. It is technician and user's responsibility to use the documents appropriate to the products used. Hardware or Firmware revisions less than 1.00 (for example 0.24) indicate the prototype products.

LAM Technologies reserves the right to modify at any moment the present document without obligation to give prior notice. This includes, for example, but not limited to, diagrams, images, organization of chapters, technical specifications of the product, features, warranty, etc.

The information contained herein replace any previously issued document.

The manual has been compiled with the intention to make it clear and complete. LAM Technologies, in order to continuously improve its products and documents quality, will appreciate any suggestion, be in change, addition or else.

1.4 Contents of the pack

The device is supplied with all connectors and ready to be mounted on DIN rail.

Technical documentation and software can be downloaded from the website www.lamtechnologies.com or may be required writing to support@lamtechnologies.com.

1.5 Safety and use conditions

This manual is intended for technicians specialized in automation or similar disciplines. In case the arguments, the terms, or the concepts expressed should not be clear you can contact our technical support writing to support@lamtechnologies.com. It is prohibited to use the products herein described if you are not sure to have understood their features and how to use and install them.

ATTENTION

The following are safety warnings and practices of primary importance that need to be fully understood and applied by the user. The user who does not fully understand the content below, or was not able to apply it totally, should not use the product for any reason.

The devices described in this manual are components. The user is responsible of the installation and use of the product that must be used only if in compliance with the rules and regulations in force. Furthermore, the user must have the technical skills needed to fully understand the features, the setting parameters and the instructions given herein. The user must also apply all the laws and specific rules of the Country and/or application in which the product is used.

The user must make the drive housing inaccessible when the drive is powered on. The user must also consider that, because of the capacitors inside the drive, it is necessary to wait at least 30 seconds from the power off before accessing the drive. According to the external capacitors eventually mounted on the power supply circuit, it is possible that the wait time is considerably longer.

During operation the product generates heat that can raise the temperature of certain parts (the heat sink for example, but not only) to values which can cause burns. Such condition persists for a long time even after the product has been turned off. The user must provide protections and appropriate warnings as well as instructing the user, the technical support and maintenance staff. The user must also describe this condition in the service manual of the finished product.

The high performance drive is able to generate strong accelerations, with high motor torque. It is therefore essential to never touch the mechanical parts with the drive powered on. The user must prepare the application so that this condition is always granted.

Because of an incorrect wiring, incorrect configuration or else, the drive can command to the motor unexpected movements. Before supplying the drive, assure that an unexpected movement of the motor does not represent danger for people, animals and things.

The power supply of the product must be isolated from the mains supply (for example through a transformer). In series to the power supply circuit, the user must always provide a protective fuse.

In normal working conditions, many control signals are isolated from the power supply; however consider that, under fault conditions, these lines can reach the same potential of the power supply and it is therefore necessary to design the application giving attention to this eventuality.

The EMC interferences can cause unexpected behavior in the whole application, therefore it is essential to minimize the spread of the EMC interferences with the use of a shielded cable, through a correct connection of the shields and of the equipotential points, etc. Furthermore, it is important to carry out a complete commissioning test after installation is complete.

The product could be permanently damaged by corrosive substances (such as gas, salts, etc.), liquid or corruptive dusts. Even a long and strong exposure to strong vibrations can cause its damage.

In some fault conditions, the drive can start sparks and fire. The housing and the components placed nearby the drive must be chosen to tolerate this eventuality and to avoid the spread of fire.

The products must never be used in explosive atmospheres (Ex areas).

The products must not be used in life support application or where the failure of the product, even in part, can cause death or damage to people, animals or things, or cause economic loss. The user not able to ensure this condition should not use the products described in this manual.

Do not dismantle the product, do not try and repair it and do not modify it unless expressly authorized by LAM Technologies.

Failure to follow the indications included in this manual can cause permanent damage to the product. For example, to power supply the product with voltage higher than the maximum one allowed, to invert the polarity of the same, to connect or disconnect the motor with the drive enabled, etc. are cause of permanent damage.

Even if the product has been designed and manufactured with extreme care, there is always the possibility that in certain circumstances it may malfunction. Therefore, for no reason, the products described in this manual can be used in life support application and in all those cases in which the unexpected failure of the product could be cause of death or damage to people, things, animals or cause economic loss.

LAM Technologies reserves the right to make changes without prior notice to the products including design, technical specification, manufacturing process and functionality.

LAM Technologies expressly declines any responsibility for any damage, whether direct or indirect, arising from the use of these products. The user who disagrees with the *user conditions* of the products, should not use them.

1.6 Warranty

LAM Technologies warrants the products described in this manual against defects in materials or workmanship for a period of 12 months. This warranty does not apply to defects, damages caused by improper use, incorrect installation or inadequate maintenance. This warranty does not apply in case the products are received modified or integrated with other parts and/or products not expressly authorized or provided for by LAM Technologies. This warranty does not apply also in case the product's label has been removed or modified.

Any request for assistance must be sent to the purchase source of the product. In case of direct purchase from LAM Technologies, a returned material authorization number (RMA) must be obtained, before shipping the device, from support@lamtechnologies.com clearly specifying the product's code, the serial number, the problem found and the assistance required. The RMA number must be clearly written on each shipping document otherwise the parcel could be rejected. The customer shall be responsible for the packaging and shipping of the defective product to LAM Technologies and shipment must be made charges prepaid. The product inspected, repaired or replaced will be available to be collected at LAM Technologies'. In case of repair under warranty LAM Technologies can, at its own discretion, repair or replace the product. No cost for material or service will be charged in case of repair under warranty.

The above warranty does not apply to the software. LAM Technologies shall not be liable for any direct or indirect damages such as, but not limited to, costs of removal and installation, lost profits, deriving from the use or the impossibility to use the software. The user who disagree with or cannot accept what stated herein, should not use or install the software.

2 Installation

The DDS5 Series drives are components. The user is responsible for the installation and use of the product that must be used only if in compliance with the rules and regulations in force. Furthermore, the user must have the technical skills needed to fully understand the features, the setting parameters and the instructions given herein.

The user must apply all the laws and specific rules of the Country and/or application in which the product is used.

The installation must be performed by expert staff and after having read and understood the instructions included herein.

2.1 Inspection

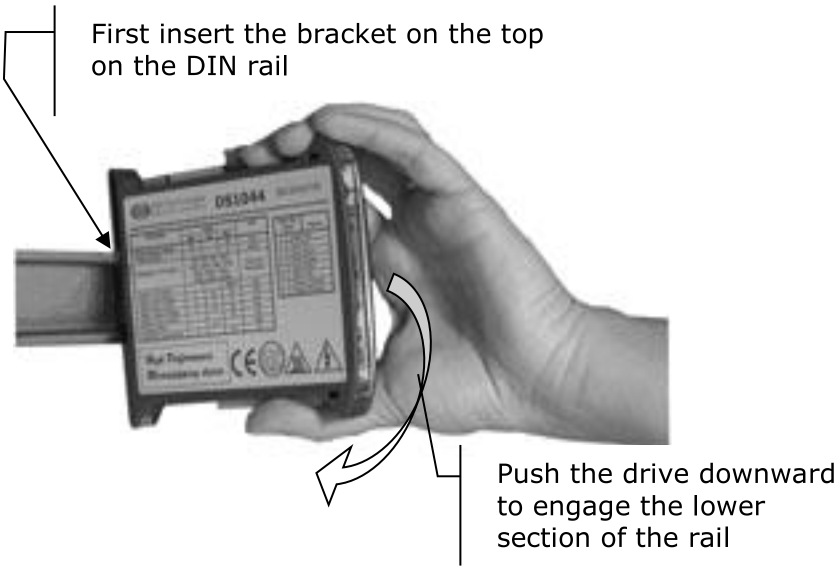
Verify that the drive is not damaged, the package is intact and all accessories are included.

Furthermore, control that the drive code corresponds to the ordered one, eventual special and customized version included. In case of problems please address to the product's vendor.

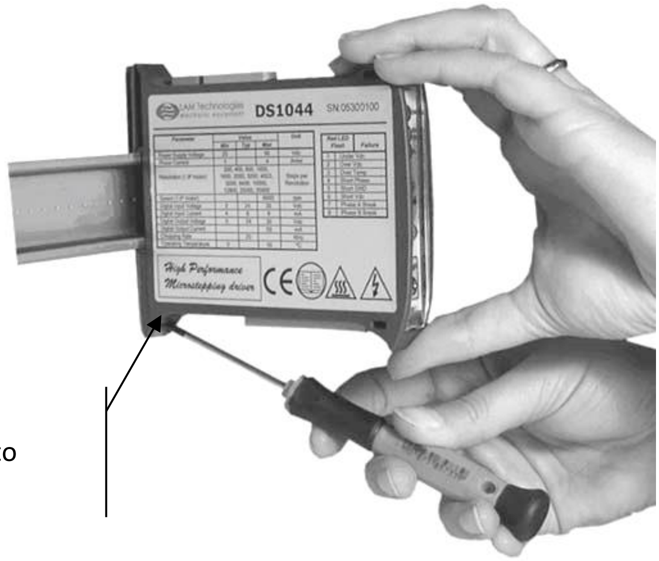
2.2 Mechanical Installation

The drive is designed to be mounted vertically on a 35mm DIN rail.

To block the drive on the DIN rail insert first the bracket on the top, on the back of the drive, over the top of the DIN rail, keeping the drive slightly inclined as shown in the picture, then push the drive downward to engage the lower section of the rail. To verify the correct engagement of the drive try and pull it slightly upward to control that it is still in position.



To remove the drive from the DIN rail insert a small flat bladed screwdriver into the hook on the bottom, on the back of the drive. Push the hook downward and pull the drive upward slightly rotating it, releasing it from the DIN rail as shown in the picture.

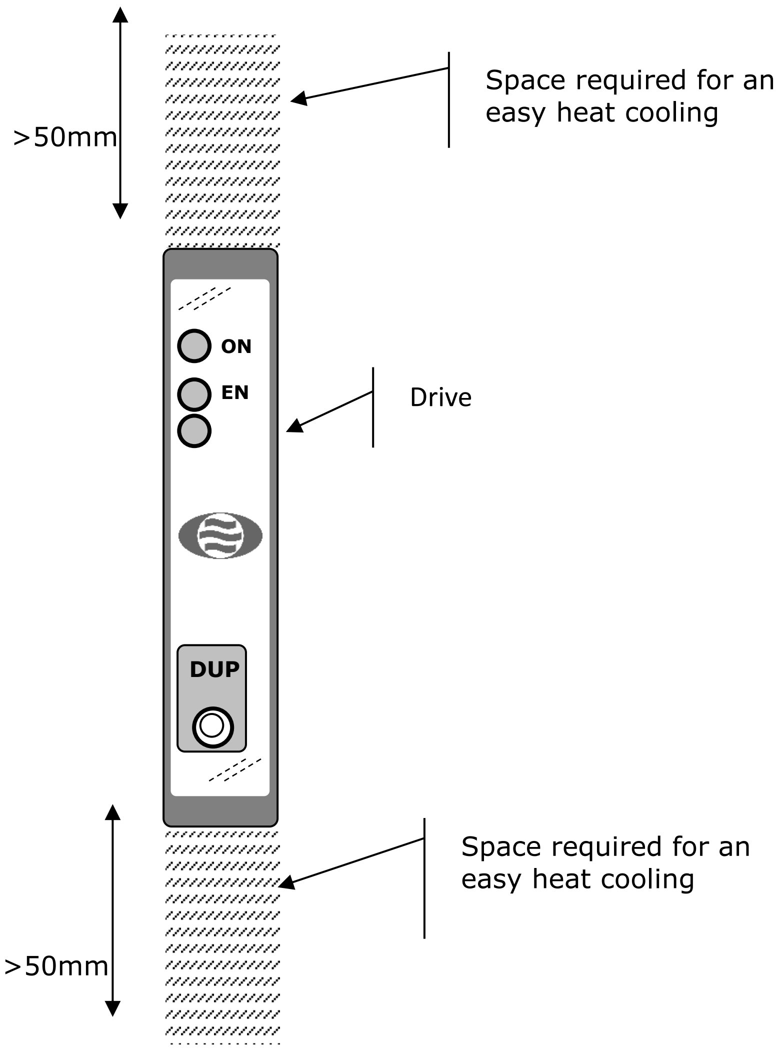


Insert a small flat bladed screwdriver into the hook

Push the screwdriver slightly downward and pull the drive upward, releasing it from the DIN rail

The heat generated by the drive while operating must be dissipated toward the surrounding air.

According to the drive calibration and to the running cycle, the space can be also substantially reduced without compromising the correct working of the drive.



2.3 Chassis setting

In order to contain the electromagnetic emissions and to better shield the drive, it is essential to give particular attention to the setting of the chassis.

The drive must be placed inside a metal case, preferably iron made, capable to successfully shield the electromagnetic emissions. The case must be electrically placed to ground as better described further below.

The filter must be correctly placed in series to the main supply. The ideal position is on the chassis edge in order to have a short wiring coming out from the main supply. If, on the contrary, the main supply cables run inside the chassis, they can be invested by electromagnetic interferences making ineffective or much reducing the filter efficiency.

The filter output earth must be connected to the metallic body of the chassis. It is important the connection is of short length and made with a large section and low inductance conductor. The point of connection between ground coming from the filter and the chassis constitutes the star center to which all other components earth must be connected. Moreover, the filter metallic body must be electrically placed in contact with the case.

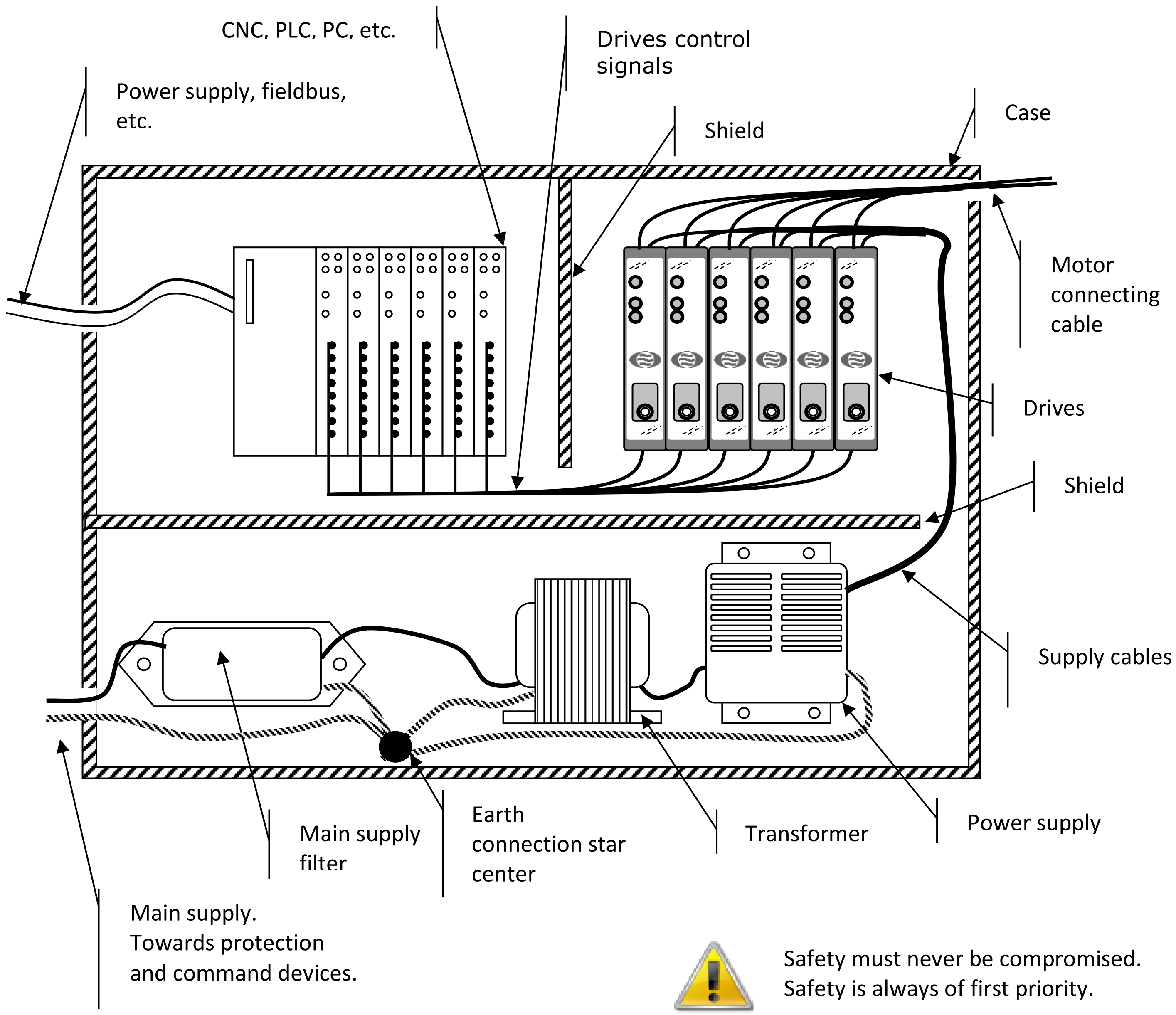
The transformer must be placed close to the filter and must have a shield between the primary and the secondary windings. The shield must be ground connected in the star center obtained inside the case. Furthermore, the transformer metallic body must be electrically connected to the case.

It is advisable to install the power supply near the transformer. If the power supply is of switching type and therefore without transformer, it is recommended to place it immediately at the output of the mains filter. The power supply earth (usually the negate pole of the filter capacitor) must be connected to ground in the star center obtained inside the case.

The drive position must be chosen in order the motor cables can immediately come out from the case without running long distances inside the case itself. Furthermore, the motor cables must be kept as far as possible from any other conductor.

The numerical control device, PLC or other, which generates the driving signals of the drive must be as far as possible from the drives and from the power supply group. Moreover, the signals wiring must be remain distant from the power supply and motor cables. When the distance from the numerical control device and the drive and/or the power supply is reduced, there must be one or more shielding walls, electrically connected to the case.

The following figure shows a possible setting of the chassis.



Safety must never be compromised.
Safety is always of first priority.

2.4 Connectors

The DDS5 drives series has 4 connectors common to the whole family and other connectors specific for the models equipped with Encoder input. The common connectors are reserved to the power supply, the motor connection, the digital I/O and *Modbus RTU*.

Connector	Function
CN1	Power supply
CN2	Motor
CN3	Digital I/Os
CN4	Encoder (only for DDS52/54 series)
CN7	Modbus RTU

The DUP port on the front part of the drive is useful for the device configuration and diagnostics which take place through the UDP30 interface and the *Omni Automation IDE* software.

Porta DUP

UDP30 interface for the device configuration and diagnostics

2.4.1 CN1 – Power Supply: AC models

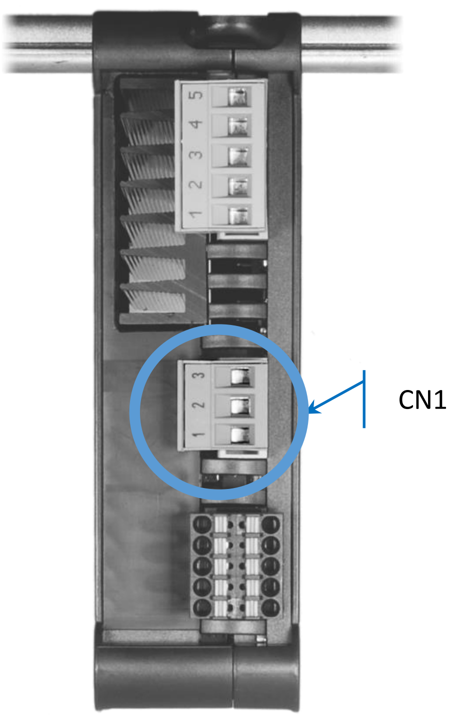
The AC supply drives are identified by a letter **A** placed at the end of the (ex. DDS5274**A**). They integrate a rectifier bridge and the capacitors necessary to rectify and filter the AC power supply voltage.

Therefore, this series of drives does not need an external power supply and can be directly connected to the output of a transformer with adequate voltage.

The overcoming of the *Vacbrk* voltage limit can permanently damage the drive.

Do not supply the drive before the wiring is complete.

Do not connect the drive with the power supply turned on.



CN1 – Power supply	
Pin	Description
1	Vac, AC power supply voltage input
2	Vac, AC power supply voltage input
3	PE, earth (internally connected to GND)

The Vac power supply must be with a sinusoidal waveform and voltage according to the values shown in the following table:

Model	Symbol	Description	Unit	Value		
				Min	Typ	Massimo
DDS5x4xA	Vac	AC supply voltage	Vac	16	32	36
	Vacbrk	Voltage causing permanent damage	Vac			42
DDS5x7xA	Vac	AC supply voltage	Vac	20	48	65
	Vacbrk	Voltage causing permanent damage	Vac			75

The drive has protections that intervene when the power supply voltage has a value that no longer guarantees correct operation.

Model	Symbol	Description	Unit	Value		
				Min	Typ	Massimo
DDS5x4xA	Vacl	Under voltage protection intervention threshold	Vac		15	
	Vach	Over voltage protection intervention threshold	Vac		38	
DDS5x7xA	Vacl	Under voltage protection intervention threshold	Vac		18	
	Vach	Over voltage protection intervention threshold	Vac		68	

For the connection with the transformer it is necessary to use a conductor with section adequate to the drive's current calibration (for safety's it is better to use the max current supplied by the drive). The following table resumes the cable section suggested for each drive:

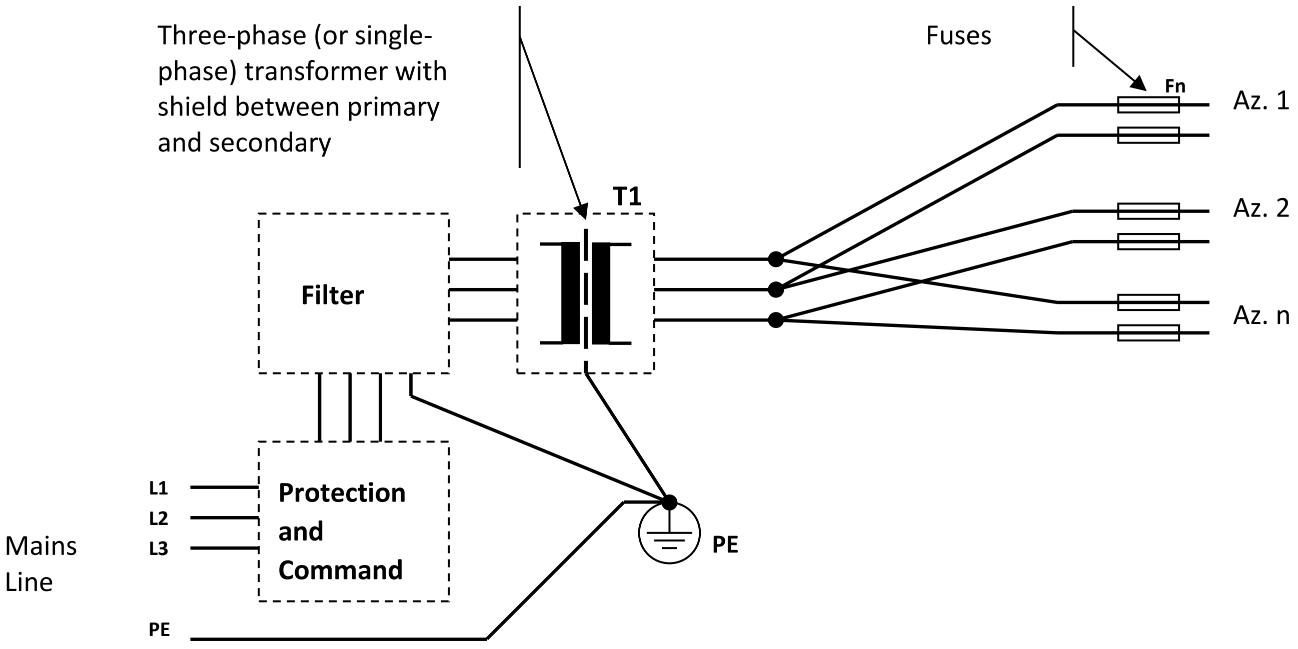
Model	Section (mm^2)
DDS5x41A	1
DDS5x44A	1
DDS5x48A	1.5
DDS5x74A	1
DDS5x78A	1.5

The power supply cable can be installed together with the ones connecting the drive to the motor. It is recommended not to place the power supply cable nearby the signal ones.

While choosing the transformer secondary voltage it is important to take into account the maximum mains fluctuation expected in the worst operative conditions, the maximum open load voltage and the minimum full load voltage and to ensure that the maximum and minimum values, results of the combination of these components, are within the maximum and minimum voltage values specified for the chosen drive model.

The power that the transformer must handle is given by the one absorbed by the load (dependent therefore on the torque required to the motor as well as the rotation speed), by the efficiency of the motor and the efficiency of the drive.

The following is an example of base connection.



The above scheme includes a three-phase transformer (note the distribution of the drives on the three phases). If necessary it is also possible to use a single-phase transformer.

Also note that the wiring must be star-like, where the earth connections of the various components ends in one only point electrically connected to the metal chassis and the earth of the plant.

Do not connect the secondary of the transformer to earth otherwise there is the risk to permanently damage the drive.

As shown in the scheme, it is necessary to put in series to the transformer primary winding a filter able to stop the emissions coming from the drive and/or present on the mains supply. Furthermore, the filter must be able to support the maximum power required by the drive plus the transformers losses.

The noise reduction level that the filter must guarantee can vary according to the laws applied to the field to which the application and/or installation belong.

The producers of filters SHAFFNER and CORCOM can represent a good reference to find the right filter.

It is mandatory to provide on each phase of the transformer primary winding a fuse able to intervene in case of short circuit or malfunctioning. It is also mandatory to use a fuse on each drives' power supply conductor

The following table shows the suggested value for some components according to the number of drives present in the application.

The calculation considers also an oscillation of the main supply voltage included in the +10/-20% range.

Model	Fuses Fn (A rit.)	Number of drives	Secondary T1 (Vac)	Power T1 (VA)	Current D1 (Arms)
DDS5x41A	2	1	32	50	25A
		2		100	25A
		3		150	25A
		4..5		250	25A
		6..8		350	25A
DDS5x44A	6,3	1	32	125	25A
		2		250	25A
		3		375	25A
		4..5		600	25A
		6..8		900	35A
DDS5x48A	12,5	1	32	250	25A
		2		500	25A
		3		750	25A
		4..5		1100	35A
		6..8		1800	50A
DDS5x74A	8	1	48	300	25A
		2		600	25A
		3		900	25A
		4..5		1400	35A
		6..8		2100	50A
DDS5x78A	16	1	48	400	25A
		2		800	25A
		3		1200	25A
		4..5		1800	35A
		6..8		2800	50A

The working voltage of the T1 transformer primary winding must be chosen according to the mains supply voltage available in the installation. The transformer must have a shield between primary and secondary windings which must be connected to earth with a short and not inductive connection. The secondary winding voltage is meant without load, with the primary winding supplied at the nominal voltage.

In the configurations with more than one drive, if the drives are not all set to the maximum current and/or if the working cycle is not simultaneous, the power of the transformer can be considerably reduced. In some cases this can also be made when the motors' speed is limited.

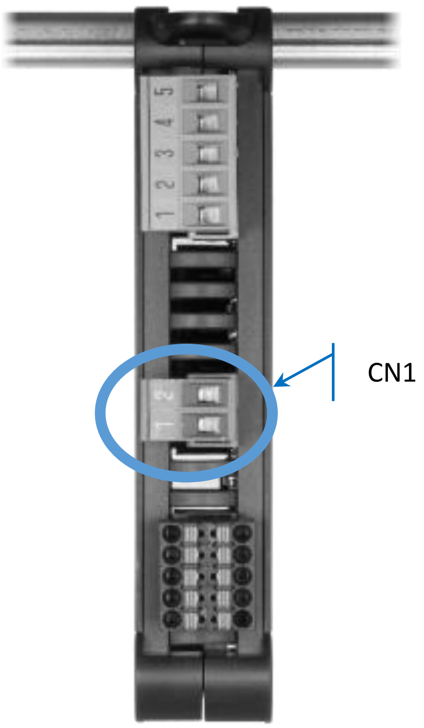
The set composed by the filter and the transformer must be used only to supply the power voltage to the drives. It is recommended not to derive other supplies from any of these parts. On the contrary, it is suggested to obtain auxiliary supplies using directly the main supply upstream of the filter.

2.4.2 CN1 – Power Supply DC models

Reverse polarity connection can permanently damage the drive as well as the exceeding of the Vpbrk voltage limit.

Do not supply the drive before the wiring is complete.

Do not connect the drive with the power supply on.



CN1 – Power Supply	
Pin	Description
1	+Vp, positive DC supply voltage
2	-Vp (GND), negative DC supply voltage

The VP power supply must be supplied according to the values specified in the following table:

Model	Symbol	Description	Unit	Value		
				Min.	Typ	Max
DDS5x4x	Vp	DC supply voltage	V	20	48	50
	Vpbrk	Voltage causing permanent damage	V	-0.5		60
DDS5x7x	Vp	DC supply voltage	V	24	72	90
	Vpbrk	Voltage causing permanent damage	V	-0.5		105

The drive has protections that intervene when the power supply voltage has a value that no longer guarantees correct operation.

Model	Symbol	Description	Unit	Value		
				Min.	Typ	Massimo
DDS5x4x	Vpl	Under voltage protection intervention threshold	V		18	
	Vph	Over voltage protection intervention threshold	V		52	
DDS5x7x	Vpl	Under voltage protection intervention threshold	V		22	
	Vph	Over voltage protection intervention threshold	V		96	

If the distance between the drive and the power supply is more than 2m, it is necessary to place near the drive (less than 10cm) an electrolytic capacitor whose minimum characteristics are specified in the following table:

Model	Voltage (V)	Capacity (μF)
DDS5x41	63	470
DDS5x44	63	470
DDS5x48	63	1000
DDS5x74	100	470
DDS5x78	100	1000

To connect the power supply, the drive and the eventual local capacitor it is necessary to use a conductor with section adequate to the current setting of the drive (for security's reason it is better to use the max rated current value of the drive). The following table resumes the cable section suggested for each drive:

Model	Section (mm^2)
DDS5x41	1
DDS5x44	1
DDS5x48	1.5
DDS5x74	1
DDS5x78	1.5

The power supply cable can be installed together with the ones which connect the drive to the motor. We recommend avoiding placing the power supply cable near the signal ones.

The power supply can be regulated or unregulated type

The use of a regulated power supply ensures a constant output voltage, immune to mains line fluctuations, and this allows to supply the drive with voltage values near to the allowed maximum ones with an immediate benefit in terms of torque supplied by the motor at high speed. The disadvantage of the regulated power supplies is their cost.

An unregulated power supply is cheaper but it forces to consider a safety's tolerance in sizing so that, in case of mains line and load fluctuations, voltage remains however within the allowed operating limits.

A detailed description of the sizing of the power supply is outside of this manual. The user who decides to assemble his own power supply must be technically qualified to size it, to ensure the correct working and to fulfill each safety requirements. To determine the power supply output voltage it must be considered the maximum mains line voltage fluctuation expected in the worst operative conditions, the maximum open load voltage and the minimum full load voltage, and to ensure that the maximum and minimum values resulting from the combination of these components are within the range of the maximum and minimum voltage specified for the chosen drive model.

The power that the power supply must deliver is given by the one absorbed by the load (thus depending from the torque required to the motor as well as from the rotation speed) and by the motor and drive efficiency.

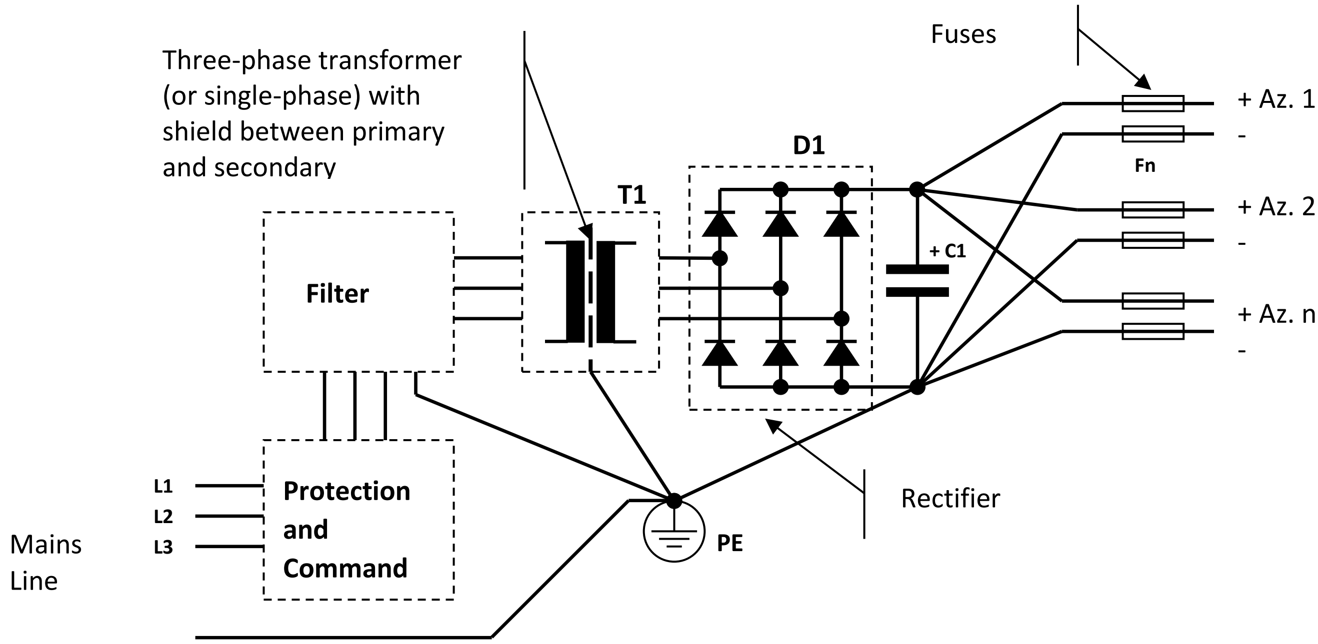
The following formula provides a rough indication:

$$Pw = 5 + (1.1 * (Iph * Iph * Rph)) + ((Vrpm * Tnm) / 7)$$

Where Pw is the power required by the power supply expressed in Watt (W), Iph is the phase current delivered to the motor expressed in effective Ampere (Arms), Rph is the motor phase resistance expressed in ohm (Ω), Vrpm is the rotation speed in rev/minute (RPM) and finally Tnm is the resistant torque of the load expressed in newton/meter (Nm). If, for example, the motor has a phase resistance of 1.5Ω and is supplied with a current of 3Arms and works at a speed of 500rpm with a load of 2Nm, the power supply should deliver a power of about 163W ((5 + (1.1*3*3*1.5) + |(500 * 2 / 7)|). Note that during the acceleration and deceleration of the load or at the enabling of the motor the absorption may be higher. For this reason it is important the power supply has output capacitors suitable to the size of the chosen drive (see further on).

To limit the peak of current at the enabling of the motor, the drive has a function able to gradually increase the phase current up to the nominal value. The ramp time can be set through the parameter *CurrentEnableRamp_MTNSTP*.

As an example, not to be considered exhaustive nor necessarily suitable to the application, it is the following basic electric diagram of an unregulated power supply with a brief indication of the values of components.



Note that the earth connection must be star-like, where the earth connections of the various components terminate into one single point electrically connected to the electrical cabinet and the earth of the plant.

In addition, the wiring to the drives must be star-like fixing the star center on the poles of the filter capacitor C1.

It is mandatory to provide a fuse on each primary winding of the transformer, able to intervene in case of short circuit or malfunctioning. It is also obligatory to place a fuse on each drive's power supply conductor.

As shown in the diagram, it is necessary to put in series to the transformer primary a filter able to block the emissions generated by the drive and/or present on the mains line.

The reduction level that the filter must guarantee may vary a lot according to the rules applied to the field to which the application and/or installation belong. Manufacturers of SHAFFNER and CORCOM filters can be a good reference to find the suitable filters for your application.

The following table shows the characteristic values of the main components of the power supply according to the number of drives present in the application. The calculations considers also an oscillation of the mains line voltage included in the +10/-20%.

Model	Fuses Fn (A rit.)	Number of drives	Secondary T1 (Vac)	Power T1 (VA)	Current D1 (Arms)	Voltage C1 (Vdc)	Capacity C1 (μF)
DDS5x41	2	1	32	50	25A	63	1000
		2		100	25A		2200
		3		150	25A		3300
		4..5		250	25A		4700
		6..8		350	25A		5600
DDS5x44	6.3	1	32	125	25A	63	3300
		2		250	25A		4700
		3		375	25A		5600
		4..5		600	25A		8200
		6..8		900	35A		10000
DDS5x48	12.5	1	32	250	25A	63	4700
		2		500	25A		6800
		3		750	25A		8200
		4..5		1100	35A		10000
		6..8		1800	50A		15000
DDS5x74	8	1	48	300	25A	100	2200
		2		600	25A		3300
		3		900	25A		3900
		4..5		1400	35A		4700
		6..8		2100	50A		6800
DDS5x78	16	1	48	400	25A	100	3300
		2		800	25A		4700
		3		1200	25A		5600
		4..5		1800	35A		8200
		6..8		2800	50A		10000

The capacity values suggested for C1 can also be obtained placing more capacitors in parallel amongst them. Eventual approximation must be made in excess. In parallel with the capacitor C1 it is recommended to place a resistor, sized appropriately, to ensure the discharge of the capacitor when the power supply is turned off.

The working voltage of the T1 transformer primary winding must be chosen according to the mains line voltage available during the installation of the application. The transformer must have a shield between primary and secondary windings which must be connected to earth by a short and not inductive connection. The secondary winding voltage is meant without load, with the primary winding supplied at the nominal voltage.

The rectifier, besides supporting the maximum current required by the drive, must be able to tolerate the surge current during the C1 capacitor charge. Such current, as being essentially limited only by the internal resistor of the transformer secondary winding, usually very low, and by the wiring, can also be of elevated entity, even if of short length (it exhausts when the capacitor is charged).

Furthermore, the rectifier needs a heat sink able to maintain the temperature within the range defined by the manufacturer (usually 70°C). The working voltage of the D1 rectifier must then be chosen according to the T1 transformer secondary winding voltage, multiplied by at least 2.

In the configurations with more than one drive, if the drives are not all set to the maximum current and/or if the working cycle is not simultaneous, the power of the transformer can be considerably reduced. In some cases this can also be done when the motor's speed is limited.

The diagram and the components' values refer to a three-phase power supply. Dimensioning in a different way the components, it is also possible to realize a single-phase power supply, which is not recommended when the required power is greater than 800W.

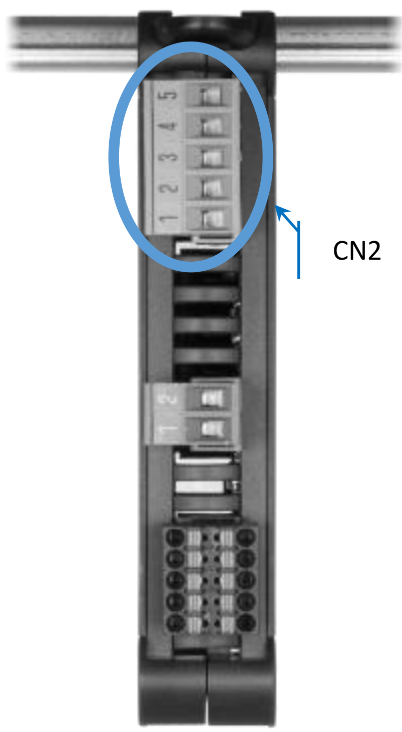
The filter, transformer and power supply unit must be used only to supply voltage to the drives. It is not recommended to derive other supplies from any of these parts. Rather, it is suggested to get auxiliary supplies using the mains line before the filter.

2.4.3 CN2 – Motor

The drive regulates the phase current of the motor modulating the power supply voltage with PWM technique. The use of a good quality shielded cable and a correct wiring are essential to better reduce the electromagnetic emission. If the cable is long, it is important to use one with low parasitic capacitance between the conductors and between the conductors and the shield.

The cable shield must be connected to the SHIELD terminal (pin 5 of CN2) of the drive but not to the body of the motor if it is electrically connected to the structure on which it is fixed. Differently, unwanted ground loop may occur which could damage the drive. Only in the case that the motor is insulated from the structure and from ground it is possible to connect the cable shield also to the motor side.

When the motor is electrically connected to the structure it is possible to connect the body of the motor to the machine ground node.



CN2 – Motor	
Pin	Description
1	A-, negate output phase A
2	A+, positive output phase A
3	B+, positive output phase B
4	B-, negate output phase B
5	SHIELD, (internally connected to GND)
Note: Inverting the phase A+ with A-, or the phase B+ with B-, the motor rotation direction is inverted.	

The cable section can be dimensioned according to the drive current setting, anyway it is suggested to choose a cable suitable to withstand the maximum current deliverable from the chosen drive.

It is also advised to connect the motor to the drive with a cable with a length inferior to 10m. For cables with a greater length, the cable size must be increased to counterbalance the voltage drop.

The following table reports the cable section suggested for each drive according to the cable length:

Model	Section (mm^2)	
	Cable length <= 10m	Cable length > 10m
DDS5x41	0.5	1
DDS5x44	1	1.5
DDS5x48	1.5	2.5
DDS5x74	1	1.5
DDS5x78	1.5	2.5

The cable connecting the drive to the motor can be installed together with the power supply cable, but it must be kept separate from the signal ones.

If you have difficulties in overcome the electromagnetic compatibility test it is possible to place in series to each phase an inductor with a value between 10uH and 100uH, and with current adequate to the set phase current. The inductor must be placed directly at the drive output.

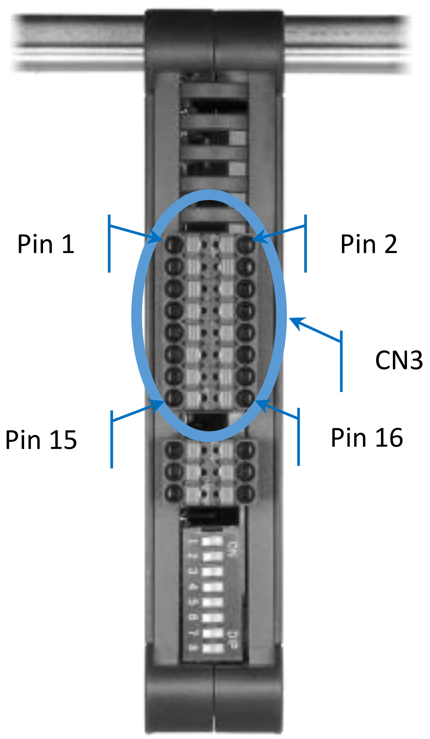
2.4.4 CN3 - I/O Control Signals

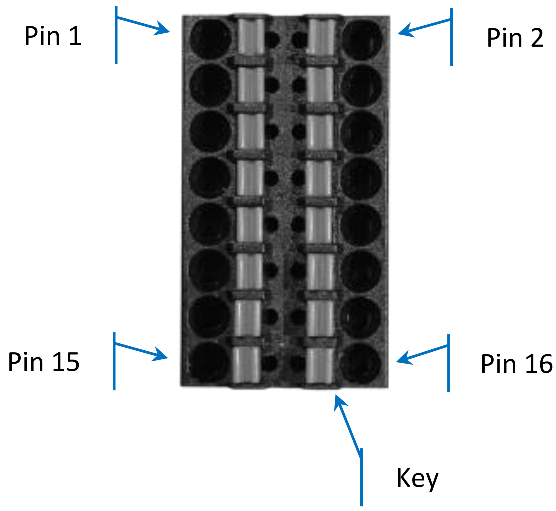
The connection with the control signals is through a 16 contacts removable spring terminal block. The terminal block can be easily oriented through the key, as shown in the picture below.

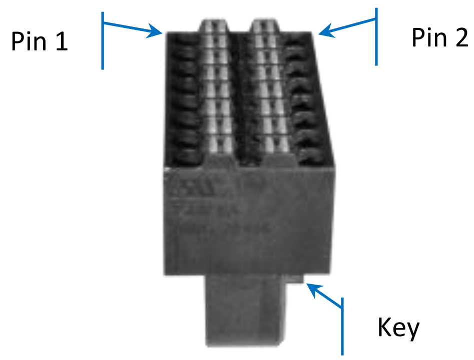
To insert the cable into the connector, press with a small screwdriver the orange presser and simultaneously insert the wire into the near hole, then release the presser. It is suggested to remove the wire covering to about 8mm.

On the CN3 connector there are a total of 6 digital inputs and 3 digital outputs.

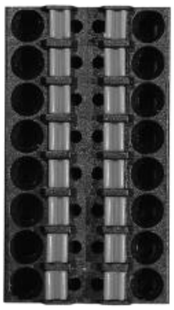
The DDS52/54 series has an additional CN5 connector (described later) for connecting the analog inputs and outputs.







The following table shows the assignment of the signals to the various terminal pins:

CN3 – Digital I/O Control Signals				
Description	Pin		Pin	Description
+24V Auxiliary Power Supply	1		2	0V (GND) Auxiliary Power Supply
DI567COM (common DI5..DI7)	3		4	DI234COM (common DI2..DI4)
DI5 (Digital Input 5)	5		6	DI2 (Digital Input 2)
DI6 (Digital Input 6)	7		8	DI3 (Digital Input 3)
DI7 (Digital Input 7)	9		10	DI4 (Digital Input 4)
DO0+ (Digital Output 0)	11		12	DO0- (Digital Output 0)
DO1+ (Digital Output 1)	13		14	DO1- (Digital Output 1)
DO2+ (Digital Output 2)	15		16	DO2- (Digital Output 2)

Note: The numbering of the inputs starts from 2 instead of 0 for coherence with other drives of the DDS series (for example DDS1), where digital inputs DI0 and DI1 have special properties.

2.4.4.1 Auxiliary Power Supply

The auxiliary power supply is optional and, if provided, allows to maintain supplied the control section of the drive, even if the power supply is removed (for example to secure the application).

Keeping supplied the logic section of the drive, the signals, the fieldbus and the encoder are maintained active. The encoder allows to keep track of the motor position change even if disconnected and moved manually.

The auxiliary power supply must be within the range shown in the table below:

Symbol	Description	Unit	Value		
			Min	Typ	Max
V24	Auxiliary Power Supply DC voltage	V	20		35
V24brk	Permanent damage voltage	V	-0.5		40

2.4.4.2 Digital Inputs

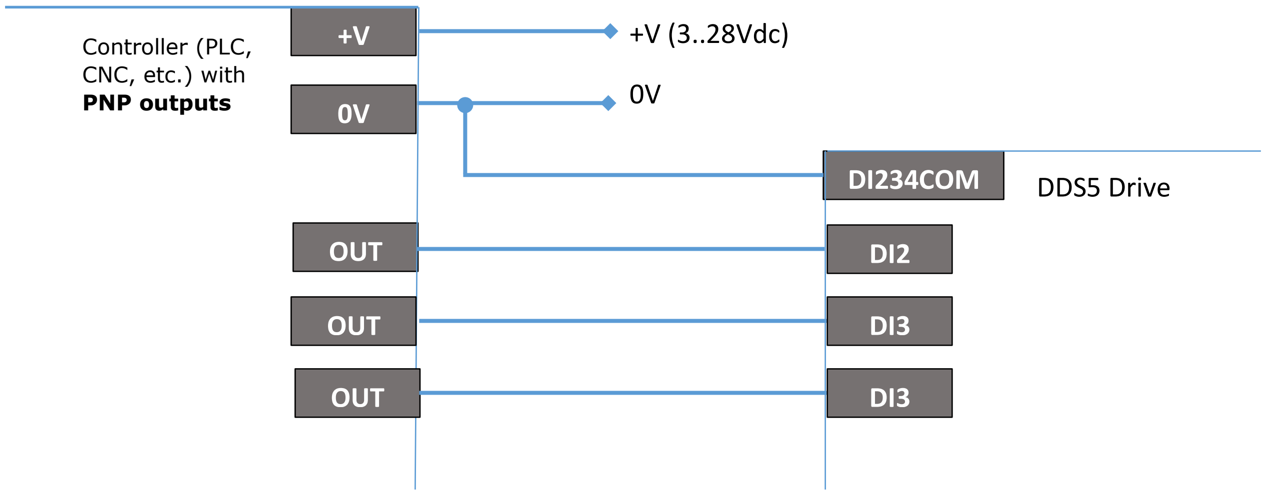
All the digital inputs are optocoupled and have a current limiting circuit which maintains the absorption constant independently from the voltage applied to the input. This allows proper operation with a wide range of input voltage without the need to add any current limit resistor. The current in fact remains constant independently from the voltage applied to the input. This simplifies the installation and the wiring.

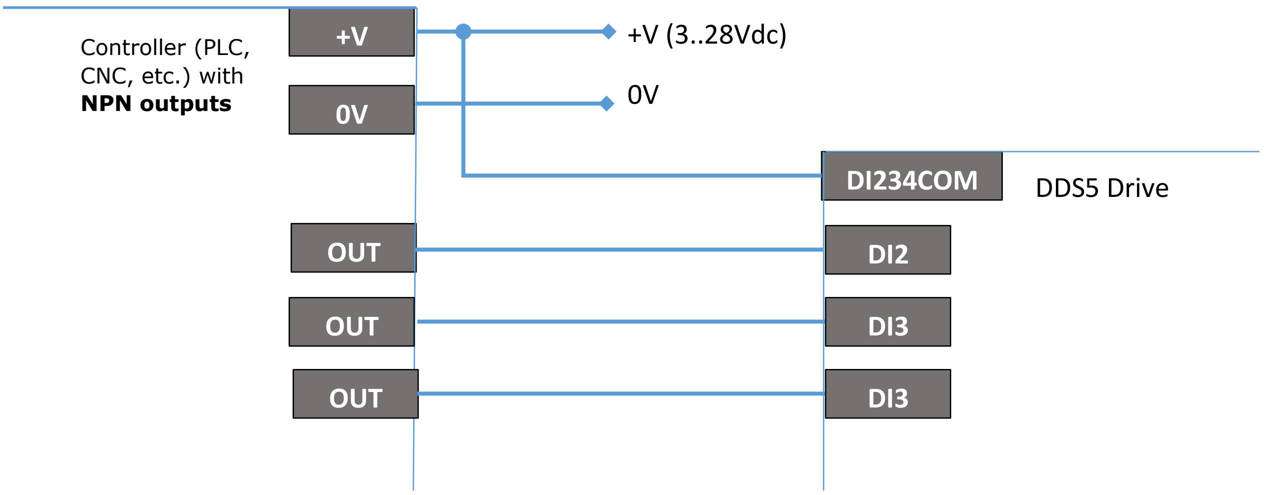
The following table shows the voltage values which correspond to the *Active* and *Inactive* input status, together with other parameters:

Symbol	Description	Unit	Value		
			Min	Typ	Max
Vdi	Active input voltage	Vdc	3		
Vdioff	Inactive input voltage	Vdc			1
Vdibrk	Digital inputs breakdown voltage	Vdc	-30		+30
Idi	Current absorbed by the digital inputs (24Vdc)	mA		5	

The inputs are organized in two groups of 3 inputs each with a common and can be used both in NPN and PNP configuration. To use a group in NPN configuration simply connect the common of the group to the positive reference and each input to the output of the master controller (PLC, CNC, etc.), while in case of PNP connection connect the common of the group to the GND and each input to the output of the master controller.

The described NPN and PNP connections are shown in the below images:





2.4.4.3 Digital Outputs

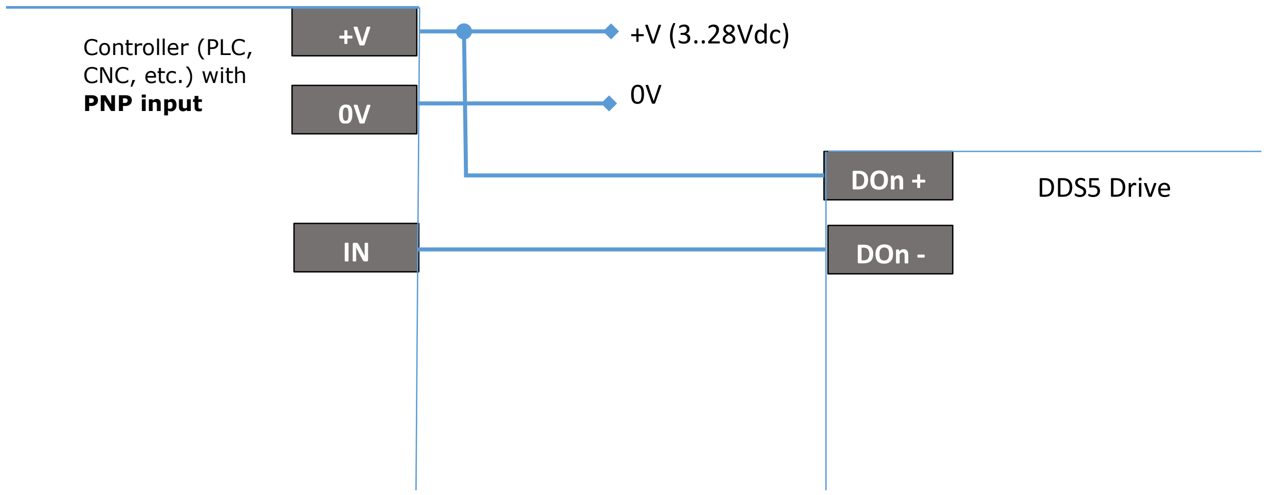
All the outputs are optocoupled and have both connections (+ and –), therefore they can be freely used in NPN and PNP configuration. On each output is placed a zener diode which allows the connection of medium entity inductive loads (for example signal relays) without the need to add an external recirculation diode.

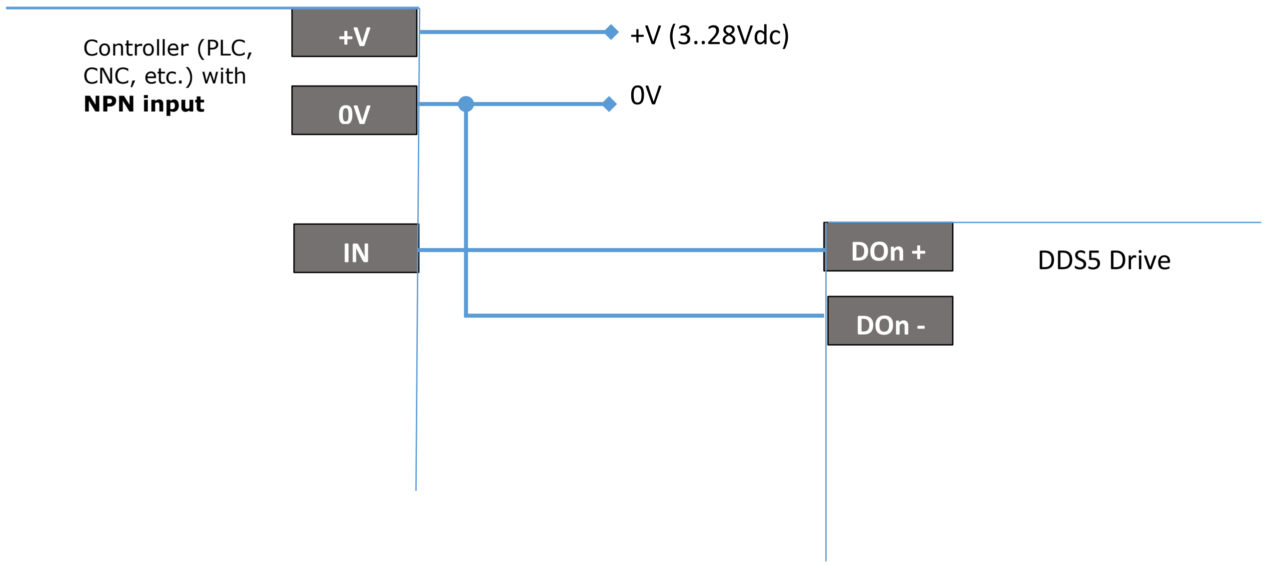
The following table shows the electrical characteristics of the digital outputs:

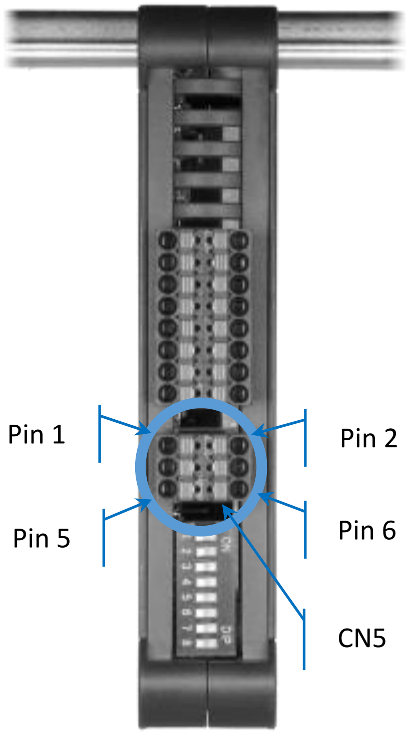
Symbol	Description	Unit	Value		
			Min	Typ	Max
Vdo	Digital output operating voltage	Vdc	1		30
Vdobrk	Digital output breakdown voltage	Vdc	-0.5		37
Vdoz	Zener diode voltage placed in parallel to each output	Vdc	37	39	42
Ido	Digital output available current	mA			80
Idobrk	Digital output breakdown current	mA	120		
Pwdo	Digital output dissipable power	mW			400

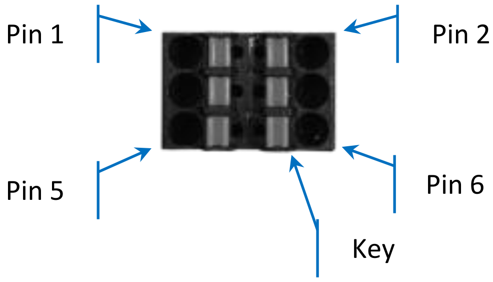
To use an output in NPN configuration simply connect the - to GND and the + to the input of the master controller (PLC, CNC, etc.), while in case of PNP connection connect the + to the positive reference and the – to the input of the control system.

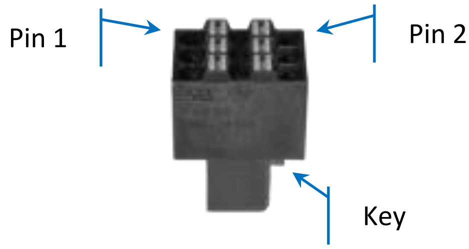
The described NPN and PNP connections are shown in the below images:





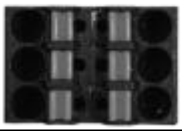






The CN5 connector is present only on the DDS52/54 series and makes 2 analog inputs and 2 analog outputs available.

The following table shows the assignment of the signals to the various terminal pins:

CN5 – Analog I/O Control Signals				
Description	Pin		Pin	Description
AO0 (Analog Output 0)	1		2	AI0 (Analog Input 0)
AO1 (Analog Output 1)	3		4	AI1 (Analog Input 1)
GND (Analog Ground)	5		6	GND (Analog Ground)

The analog inputs and outputs are not isolated and the ground reference of the analog signals is connected internally to the drive with the terminal 2 of CN1 (-Vp).

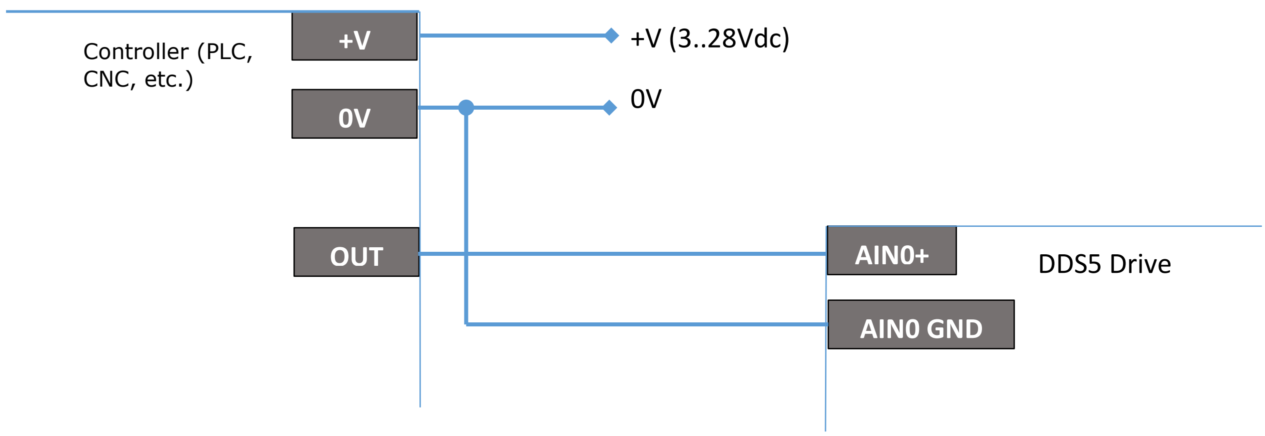
2.4.4.4 *Analog Inputs*

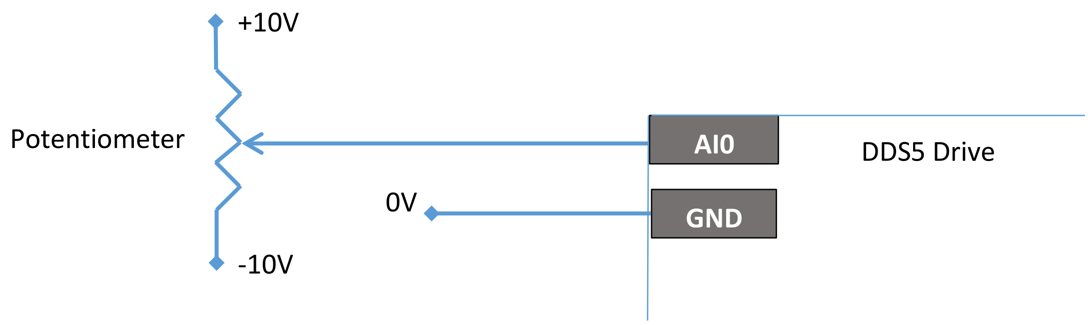
The analog input is able to measure voltage between -10V and +10V.

The following table shows the electrical characteristics of the analog inputs:

Symbol	Description	Unit	Value		
			Min	Typ	Max
Vai	Analog input operating voltage	Vdc	-10.2		+10.2
Vaibrk	Analog input breakdown voltage	Vdc	-45		+45
Rai	Analog inputs impedance	KΩ		47	
ADst	A/D converter conversion time	ms		1	
ADsoff	A/D converter start offset	%fs		1	
ADdoff	A/D converter offset drift	%fs		0.2	
ADline	A/D converter linearity error	%fs		1	

Example of connection of the analog input:





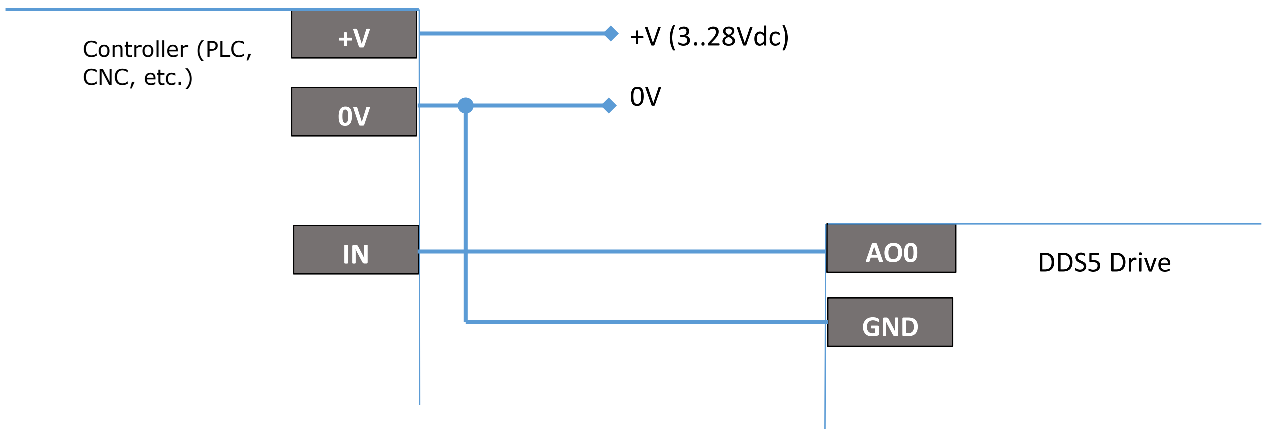
2.4.4.5 Analog Outputs

The analog outputs are able to deliver voltages between 0 and 10V.

The following table shows the electrical characteristics of the analog outputs:

Symbol	Description	Unit	Value		
			Min	Typ	Max
Vao	Analog outputs operating voltage	Vdc	0		+10,2
Iao	Analog outputs operating current	mA		10	
Rai	Analog outputs impedance	Ω			47
DAst	A/D converter conversion time	ms		1	
DAsoff	A/D converter start offset	%fs		1	
DAdoff	A/D converter offset drift	%fs		0.2	
DAline	A/D converter linearity error	%fs		1	

Example of connection of the analog output:

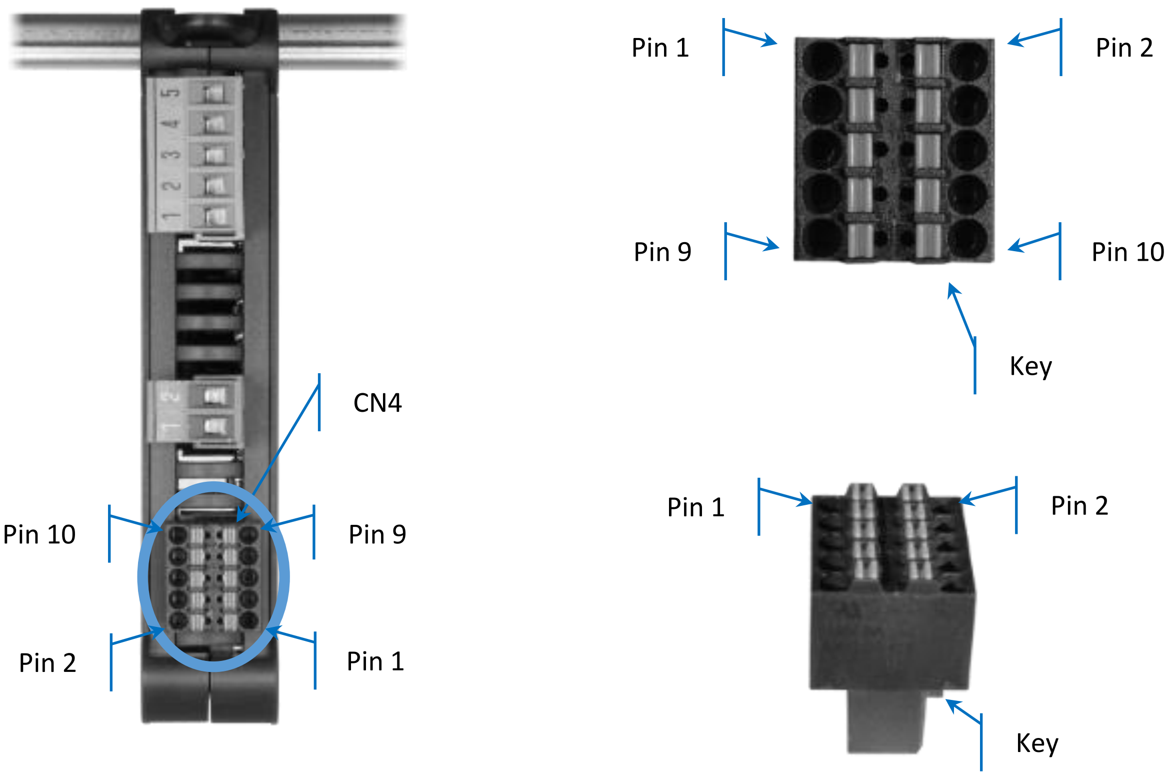


2.4.5 CN4 – Motor Encoder and SSI Encoder

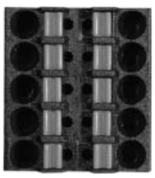
The 10 contacts removable spring terminal block CN4 is used for connecting the incremental encoder and the encoder with synchronous serial interface (SSI - Synchronous Serial Interface) . The terminal block can be easily oriented through the key, as shown in the picture below.

The CN4 connector is present only on the DDS52/54 Series drives.

To insert the cable into the connector, press with a small screwdriver the orange presser and simultaneously insert the wire into the near hole, then release the presser. It is suggested to remove the wire covering to about 8mm.



The following table shows the assignment of the signals to the various terminal pins:

CN4 – Motor Encoder and SSI Encoder				
Description	Pin		Pin	Description
+V Encoder Power Supply	1		2	**0V** (GND) Encoder Power Supply
A+ Incremental Encoder	3		4	**A-** Incremental Encoder
B+ Incremental Encoder	5		6	**B-** Incremental Encoder
I+ (Index) Incremental Encoder / **Data+** SSI Encoder	7		8	**I-** (Index) Incremental Encoder / **Data-** SSI Encoder
Clock+ SSI Encoder	9		10	**Clock-** SSI Encoder

The *I (Index)* and *Data* signals must be connected one as an alternative to the other. It follows that when using the SSI encoder the Index signal of the incremental encoder cannot be used.

For the connection between the drive and the encoder, it is suggested to use a shielded cable, having care to connect the shield on the Pin2 together with the 0V reference.

To power the Encoder, the drive supplies a voltage of +5V with a current of 100mA suitable for the most encoders, however it is also possible to connect encoders with a different supply voltage, provided that they are externally supplied.

2.4.5.1 Incremental Encoder

It is possible to use any incremental encoder with or without Index (also called zero mark) provided that it has a resolution within the configuration's (register *CPR_ENCMTR*).

The signals inputs A, B and I are Line Driver type and usually they do not require terminating resistors. The drive internal circuits also allow the connection of other signals types, as shown in the table below:

Encoder output signals types	Encoder Signal	Drive Signal	Notes
Line Driver	A+	A+	
	A-	A-	
	B+	B+	
	B-	B-	
	I+	I+	
	I-	I-	
TTL/CMOS	A	A+	The inputs A-, B- and I- remains disconnected.
	B	B+	
	I	I+	
Open Collector	A	A+	The inputs A-, B- and I- remains disconnected.
	B	B+	
	I	I+	
Push-Pull	A	A+	**ATTENTION, when using an encoder supplied with voltage higher than 5V with push-pull outputs, it is important to insert in series to each signal a diode (1N4148 for example) with the cathode facing the encoder and the anode connected to the drive, otherwise the drive itself could be damaged.**
	B	B+	
	I	I+	

2.4.5.2 SSI Encoder

It is possible to connect encoders with SSI interface with binary coding and data frame length between 8 and 31 bits. If the frame is longer, the bits after 31 will not be read.

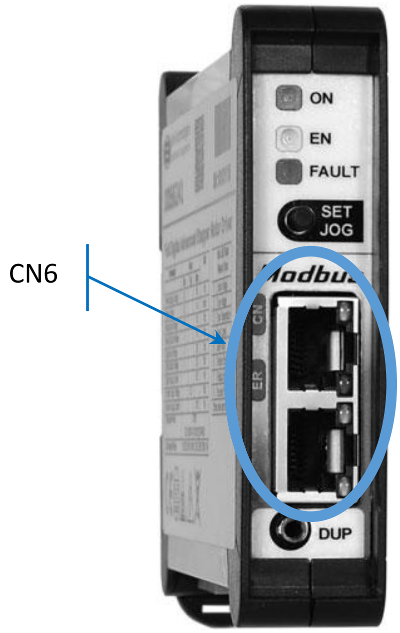
The *Clock* and *Data* signals are of Line Driver type and normally do not require termination resistance.

The following table shows the main features of the SSI interface implemented inside the drive.

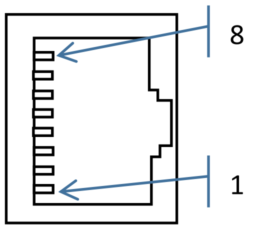
Symbol	Description	Unit	Value		
			Min	Typ	Max
SSIclk	Clock frequency	MHz		1.0	
SSIfb	Frame length	bit	8		47
SSImtb	Number of bits for full revolutions	bit	0		24
SSIstb	Number of bits for single revolution	bit	8		31
SSIcd	Position coding			Binary	

2.4.6 CN6 – Modbus RTU

The CN6 connector has two standard 8-pole RJ45 sockets and is used for connection to the Modbus RTU bus. All the signals of each socket are connected together.



The following table shows the correspondence of signals to the connector's pins:

CN6 – Bus Modbus RTU		
Description	Pin	 Socket front view
RS485_GND	8	
RS485_VCC	7	
Not used	6	
RS485_D1 (B, RxTx-)	5	
RS485_D0 (A, RxTx+)	4	
Not used	3	
Not used	2	
Not used	1	

2.4.6.1 LED

The CN6 connector has two LEDs that indicate the presence of data traffic and errors.

Led *CN (green)*	
Off	No data traffic
On	Data Frame received

Led *ER (red)*	
Off	Absence of error
Flashing	Presence of error

3 Drive Configuration

The configuration can be divided into bus configuration and drive configuration. The first one is necessary to allow the exchange of data according to the Modbus RTU protocol, while the second one is used to adapt the drive to the application.

The bus configuration involves the setting of the note address, the format and the communication speed and is done using the free software *Omni Automation IDE* described further below.

The drive configuration is obtained by writing appropriate values in the registers and can be done through the Modbus RTU communication or through *Omni Automation IDE*. If desired, the registers can be saved in the non-volatile memory of the device to be automatically reloaded at the following power on.

By saving the value of a register it is possible to adapt the device to the application without having to configure it every time. Apparently this seems like a simplification but it requires to prepare the device (saving the desired values in the registers) before being able to use it in the application. When there are many applications, or are updated over time, you have to keep a record with the values of the registers used in each applications and in each version and this can become complex and give rise to errors. On the contrary, by having the master configure the device at each power on, it will be possible to install simply a new device without worrying about other than the configuration of the bus. In this case, in fact, the specific application will initialize the dictionary registers with the desired values and without the possibility of error. In addition, in the event that the device need to be replaced, the technical assistance can simply send a new device without having to worry about the application and the version in which it will be installed.

4 Modbus RTU

The Modbus RTU protocol is an industrial standard with millions of nodes installed worldwide. It is an open, documented protocol, that can be easily supported even by embedded systems. Most PLCs on the market equipped with an RS485 port natively implement the Modbus RTU protocol.

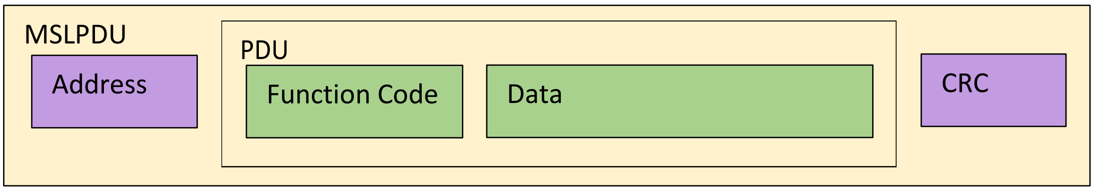
The structure is client/server type, where the server transmits data only in response to client queries. The client is represented by the control device, such as a PLC or a PC, while the server by the DDS5 drive.

The DDS5 Series drives implement the standard Modbus RTU protocol as described in the specification "MODBUS APPLICATION PROTOCOL SPECIFICATION V1.1b3" and "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

For an exhaustive description of the protocol, please refer to the official documentation while below is an introduction to the implemented functionalities and to the specific characteristics of the DDS5 Series drives.

4.1 MSLPDU (Modbus Serial Line Protocol Data Unit)

The MSLPDU block contains the node address, the PDU block and the CRC, useful to verify the integrity of the message. The PDU block (Protocol Data Unit) is in turn divided into the Function Code and the data block.



MSLPDU:

- *Address* (1 byte), useful for identifying the node
- *PDU* (variable length), it contains the protocol data
- *CRC* (2 bytes), useful for verifying the integrity of the message

4.1.1 Address

It represents the address of the server (slave device, drive) recipient of the message when the data flow is from the client (master device, PLC) to the server, or the address of the server address that transmits the response when the data flow is from the server to the client.

The drive address (the server) can be any value between 1 and 247 and must be unique across the mains line. The address setting can be done through the *Omni Automation IDE* software or through the dip-switch (in this case the maximum address is limited to 127).

The client can send direct messages to all servers in the mains line (broadcast message) using the value 0 as the destination address. All servers process the message but they do not provide any response.

4.1.2 Function code

It represents the identifier of the function that the client commands to the server., when the data flow is from the client to the server, or the function processed by the server when the data flow is from the server to the client.

4.1.3 Data

Specific data of the invoked function, transmitted from the client to the server or vice versa.

4.1.4 CRC

It represents the checksum calculated by the sender on all bytes of the message. The algorithm used is detailed in chapter *6.2.2 CRC Generation* of the document "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

4.2 Supported Services and Features

The supported services as the following :

Services	Function Code	Description
Read Holding Registers	3 (03_h)	It reads in sequence a specified number of registers starting from an address.
Write Single Register	6 (06_h)	It writes the value in a register.
Write Multiple Registers	16 (10_h)	It writes a sequence of values in a specified number of register starting from an address.
Mask Write Register	22 (16_h)	It changes the bits of a register through the AND and OR operation.
Read/Write Multiple Registers	23 (17_h)	It reads in sequence a specified number or registers starting from a reading address and writes a sequence of values in a specified number of registers starting from a writing address.

4.2.1 Read Holding Registers 3 (03_h)

It allows to read in sequence a specified number of registers starting from a specified address.

Below is an example of reading the 42811_h *StatusDWord*, 42813_h *ActualTorque*, 42814_h *ActualVelocity* and 42816_h *ActualPosition* registers of 32bit, 16bit, 32bit and 32bit size respectively. The 4 addresses are contiguous, therefore it is possible to read them with a single *Read Holding Registers* command.

The drive has been configured with address 76.

Messages related to the reading of registers 42811_h *StatusDWord*, 42813_h *ActualTorque*, 42814_h *ActualVelocity* and 42816_h *ActualPosition*	
4C 03 A7 3A 00 07 08 AC	Client (master) Direct request to the drive Byte expressed in hexadecimal
4C 03 0E 02 00 20 03 01 24 00 00 10 9D 00 3E FA E8 F1 C1	Server (slave) Drive response Byte expressed in hexadecimal

MSLPDU (Modbus Serial Line Protocol Data Unit) transmitted by the client (master)				
	PDU (Protocol Data Unit)			
4C	03	A7 3A	00 07	08 AC
Address = 76	Function code = 03_h	Starting Address = 42810	Quantity of Registers = 7	CRC = $08AC_h$

MSLPDU (Modbus Serial Line Protocol Data Unit) transmitted by the client (master)				
	PDU (Protocol Data Unit)			
4C	03	0E	02 00 20 03 01 24 00 00 10 9D 00 3E FA E8	F1 C1
Address = 76	Function code = 03_h	Byte count = 14	Registers Value *StatusDWord* = 02002003_h *ActualTorque* = 292 *ActualVelocity* = 4253 *ActualPosition* = 4127464	CRC = $08AC_h$

For more details see chapter *6.3 03 (0x03) Read Holding Registers* of the "*MODBUS Application Protocol Specification V1.1b3"* and the document "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

4.2.2 Write Single Register 6 (06h)

It allows to write a register to a specified address with a specified value.

Below is an example of writing the 42913_h *TargetTorque* register of 16bit size.

The drive has been configured with address 76.

Messages related to the writing of register 42913_h *TargetTorque*	
4C 06 A7 A0 02 A3 E4 58	Client (master) Direct request to the drive Byte expressed in hexadecimal
4C 06 A7 A0 02 A3 E4 58	Server (slave) Drive response Byte expressed in hexadecimal

MSLPDU (Modbus Serial Line Protocol Data Unit) transmitted by the client (master)				
	PDU (Protocol Data Unit)			
4C	06	A7 A0	02 A3	E4 58
Address = 76	Function code = 06_h	Starting Address = 42912	Register Value = 675	CRC = $E458_h$

MSLPDU (Modbus Serial Line Protocol Data Unit) response sent by the server (drive)				
	PDU (Protocol Data Unit)			
4C	06	A7 A0	02 A3	E4 58
Address = 76	Function code = 06_h	Starting Address = 42912	Register Value = 675	CRC = $E458_h$

For more details see chapter *6.6 06 (0x06) Write Single Register* of the "*MODBUS Application Protocol Specification V1.1b3"* and "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

4.2.3 Write Multiple Registers 16 (10h)

It allows to write in sequence a specific number of registers starting from a specified address.

Below and example of writing the 42911_h *ControlDWord*, 42913_h *TargetTorque*, 42914_h *TargetVelocity* and 42916_h *TargetPosition* registers with dimensions 32bit, 16bit, 32bit and 32bit respectively. The 4 registers are contiguous, therefore it is possible to write them with a single *Write Multiple Registers* command.

The drive has been configured with address 76.

Messages related to the writing of registers 42911_h *ControlDWord*, 42913_h *TargetTorque*, 42914_h *TargetVelocity* and 42916_h *TargetPosition*	
4C 10 A7 9E 00 07 0E 01 00 00 03 03 E8 00 00 17 70 00 04 93 E0 F0 C6	Client (master) Direct request to the drive Byte expressed in hexadecimal
4C 10 A7 9E 00 07 CC 8C	Server (slave) Drive response Byte expressed in hexadecimal

MSLPDU (Modbus Serial Line Protocol Data Unit) transmitted by the client (master)						
	PDU (Protocol Data Unit)					
4C	10	A7 9E	00 07	0E	01 00 00 03 03 E8 00 00 17 70 00 04 93 E0	F0 C6
Address = 76	Function code = 10_h	Starting Address = 42910	Quantity of Registers = 7	*Byte Count* = 14	Registers Value *ControlDWord* = 01000003_h *TargetTorque* = 1000 *TargetVelocity* = 6000 *TargetPosition* = 300000	CRC = $F0C6_h$

MSLPDU (Modbus Serial Line Protocol Data Unit) response sent by the server (drive)				
	PDU (Protocol Data Unit)			
4C	10	A7 9E	00 07	CC 8C
Address = 76	Function code = 10_h	Starting Address = 42910	Quantity of Registers = 7	CRC = $CC8C_h$

For more details see chapter *6.12 16 (0x10) Write Multiple registers* of the "*MODBUS Application Protocol Specification V1.1b3"* and the document "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

4.2.4 Mask Write Register 22 (16h)

It allows to modify the bits of a register through AND and OR operations. The result is obtained by applying the formula:

Result = (Current Contents AND *And_Mask*) OR (*Or_Mask* AND (NOT *And_Mask*))

Below is an example of changing the bits of the 42922_h *Outputs_DOVd* register.

The drive has been configured with address 76.

Messages related to the changing of the bits of register 42922_h *Outputs_DOVd*	
4C 16 A7 A9 00 04 00 02 B6 A0	Client (master) Direct request to the drive Byte expressed in hexadecimal
4C 16 A7 A9 00 04 00 02 B6 A0	Server (slave) Drive response Byte expressed in hexadecimal

MSLPDU (Modbus Serial Line Protocol Data Unit) transmitted by the client (master)					
	PDU (Protocol Data Unit)				
4C	16	A7 A9	00 04	00 02	B6 A0
Address = 255	Function code = 16_h	Reference Address = 42921	And_Mask = 4	Or_Mask = 2	CRC = $B6A0_h$

MSLPDU (Modbus Serial Line Protocol Data Unit) response sent by the server (drive)					
	PDU (Protocol Data Unit)				
4C	16	A7 A9	00 04	00 02	B6 A0
Address = 255	Function code = 16_h	Reference Address = 42921	And_Mask = 4	Or_Mask = 2	CRC = $B6A0_h$

For more details see chapter *6.16 22 (0x16) Mask Write Register* of the *MODBUS Application Protocol Specification V1.1b3"* and the document "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

4.2.5 Read/Write Multiple Registers 23 (17h)

It allows to read and write in sequence a specified number of registers starting from a specified address.

Below is an example of reading the 42811 *StatusDWord*, 42813 *ActualTorque*, 42814 *ActualVelocity* and 42816 *ActualPosition* registers with dimensions 32bit, 16bit, 32bit and 32bit respectively and of writing of 42911 *ControlDWord*, 42913 *TargetTorque*, 42914 *TargetVelocity* and 42916 *TargetPosition* registers with dimensions 32bit, 16bit, 32bit and 32bit respectively. The registers in reading and writing are contiguous and therefore it is possible to read and write them with one single *Read/Write Multiple Registers* command.

The drive has been configured with address 76.

Ethernet frames related to the reading of registers 42811 *StatusDWord*, 42813 *ActualTorque*, 42814 *ActualVelocity,* 42816 *ActualPosition* and to the writing of the registers 42911 *ControlDWord*, 42913 *TargetTorque*, 42914 *TargetVelocity* and 42916 *TargetPosition*.	
4C 17 A7 3A 00 07 A7 9E 00 07 0E 02 00 00 03 02 58 FF FF EC 78 FF FB 6C 20 85 D2	Client (master) Direct request to the drive Byte expressed in hexadecimal
4C 17 0E 02 00 20 03 02 68 FF FF F1 BD FF B0 1E 4F E6 FA	Server (slave) Drive response Byte expressed in hexadecimal

MSLPDU (Modbus Serial Line Protocol Data Unit) transmitted by the client (master)								
	PDU (Protocol Data Unit)							
4C	17	A7 3A	00 07	A7 9E	00 07	0E	02 00 00 03 02 58 FF FF EC 78 FF FB 6C 20	85 D2
Address = 76	Function code = 17_h	Read Starting Address = 42810	Quantity to Read = 7	Write Starting Address = 42910	Quantity to Write = 7	*Write Byte Count* = 14	Write Registers Value *ControlDWord* = 02000003_h *TargetTorque* = 600 *TargetVelocity* = -5000 *TargetPosition* = -300000	CRC = $85D2_h$

MSLPDU (Modbus Serial Line Protocol Data Unit) response sent by the server (drive)				
	PDU (Protocol Data Unit)			
4C	17	0E	02 00 20 03 02 68 FF FF F1 BD FF B0 1E 4F	E6 FA
Address = 76	Function code = 17_h	Byte count = 14	Registers Value *StatusDWord* = 02006003_h *ActualTorque* = 626 *ActualVelocity* = -1626 *ActualPosition* = -22035	CRC = $E6FA_h$

For more details see chapter *6.17 23 (0x17) Read/Write Multiple registers* of the "*MODBUS Application Protocol Specification V1.1b3"* and the document "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

4.3 Representation of 8bit, 16bit and 32bit registers

The Modbus protocol provides only word (16bit) read and write operations, however it is also possible to read and write also 8bit registers, using only the low byte of the word, or 32bit registers using two consecutive words.

Below is an example of how registers of different sizes are represented within the words.

32bit register at the address n containing the value 305.419.896 (12345678_h)

Word address n	Word address n+1
4660 (1234_h)	22136 (5678_h)

16bit register at the address n containing the value -26.506 (9876_h)

Word address n
-26.506 (9876_h)

8bit register at the address n containing the value 90 ($5A_h$)

Word address n
90 ($5A_h$)

The values are always transmitted and received in big-Endian” format, i.e. from the most significant byte to the less significant one.

4.4 Registers numbering and address

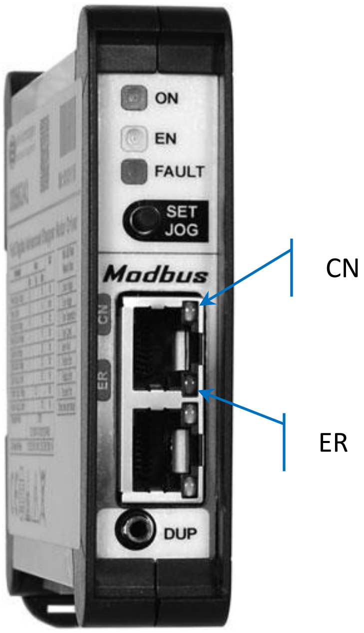
The drive registers are numbered starting from 1 and their address corresponds to their numbering, therefore, the addresses of the registers included in this manual have base 1. According to the Modbus RTU addressing rules, the registers are instead addressed starting from 0, consequently the transmitted address must be 1 less than the address in this manual.

For example, to read the register 40101 *Product code* the message will have to be compiled with the address 40100 (40101 - 1) as indicated below (drive address 76).

MSLPDU (Modbus Serial Line Protocol Data Unit) transmitted by the client (master)				
	PDU (Protocol Data Unit)			
4C	03	9C A4	00 02	A4 65
Address = 76	Function code = 03_h	Starting Address = 40100	Quantity of Registers = 2	CRC = $A465_h$

Typically the client device (PLC, PC, etc.) autonomously subtracts 1 from the address set for the reading or writing of a register, however, if there should be errors of non-existent register, it is recommended to try and insert the address indicated in the manual decremented by 1.

4.5 LED



The presence of data traffic is indicated by the green CN LED:

Led *CN (green)*	
Off	No data traffic
On	Data frame received

The red ER LED flashes when errors occur. The number of flashes indicates the type of error according to the following table:

Led *ER*	
Off	No error
4 Flashes	Client watchdog expired
6 Flashes	Timing error between characters

4.6 Errors

If the client requests a service not supported by the drive or tries to access a non-existent register, an exception occurs that generates from the drive a particular response in which 80h is added to the *Function code*, followed by the byte *Exception Code* containing the error code.

Below is the message obtained in response to an attempt to read a non-existent register with address 40000. In the example, the address of the drive is 76 and the 03_h *Read Holding Registers* function is used.

MSLPDU (Modbus Serial Line Protocol Data Unit) response sent by the server (drive)			
	PDU (Protocol Data Unit)		
4C	83	02	50 E6
Address = 76	Function code = 83_h	Exception Code = 2	CRC = $50E6_h$

An exception may occur also in other cases such as trying to write an invalid value or trying to perform an operation when the status of the drive does not allow it.

The following table contains the error codes returned to the client:

Exception Code	Name	Description
1	Illegal Function	The required service is not implemented and the corresponding *Function Code* is not recognized.
2	Illegal Data Address	It occurs when there is no register at the indicated address. Note that in case of services that access to register in sequence (such as *Write Multiple Registers*), the non-existent register could be subsequent to the first.
3	Illegal Data Value	It indicates a Modbus RTU protocol violation such as an incorrect length or invalid values.
4	Server Device Failure	It occurs when the drive encounters an error in the execution of the service as occurs for example when trying to write a read-only register, a not allowed value, etc.

For more details see chapter *7 MODBUS Exception Responses of the MODBUS Application Protocol Specification V1.1b3* and the document "*MODBUS over Serial Line Specification and Implementation Guide V1.02*" available on the official website www.modbus.org.

5 Omni Automation IDE Software

The free Omni Automation IDE software (hereinafter OAI) running under Windows (Windows 7, Windows 8.1 and Windows 10 32bit or 64bit) allows to perform device diagnostics and to read and write registers in real time.

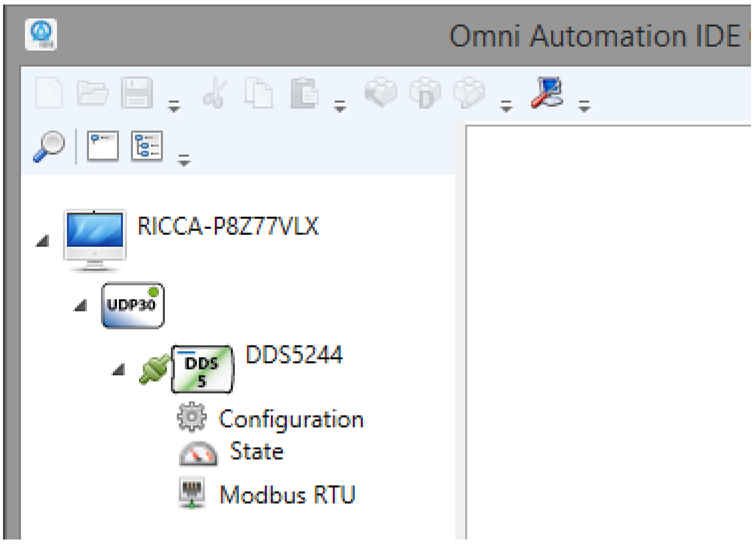
The connection between the PC and the drive is made through the UDP30 interface which also galvanically isolates the PC from the device. Furthermore, the UDP30 interface is able to power the logic section, allowing for example to update the firmware even without the power supply.

When drive powered only by the UDP30 interface receives the auxiliary power (+24V) and/or the power supply, it automatically restarts.

When the drive is powered only by the UDP30 interface, the power stage, the field bus and the I/Os (digital, analog, encoder, etc.) are not working.

After connecting the UDP30 interface to the DUP port on the front of the drive, it is possible to push the *Search* button so that OAI starts scanning the UDP30 interface and the connected devices.

When the search is complete, the tree of the devices connected to the PC appears, looking similar to the following image:

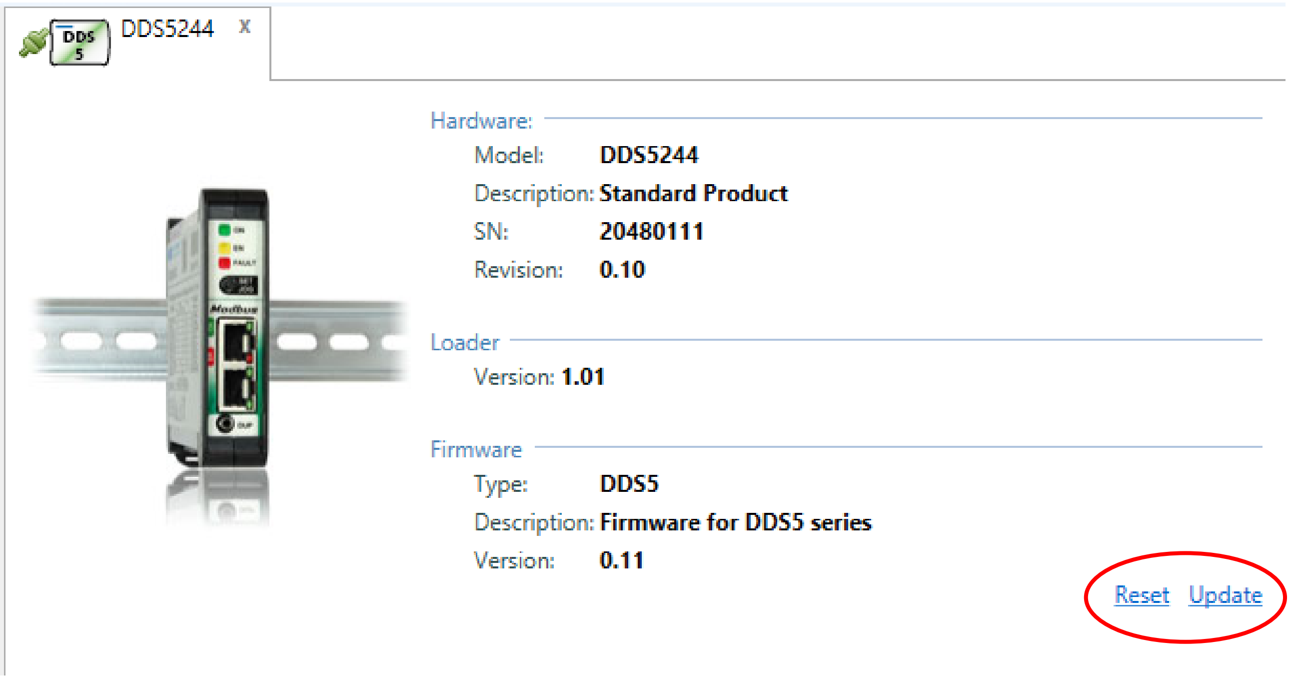


5.1 Firmware update and reset

By double-clicking on the drive's name (DDS5244 in the image) it appears the tab with the characteristic data of the device as the serial number, the firmware version, etc.

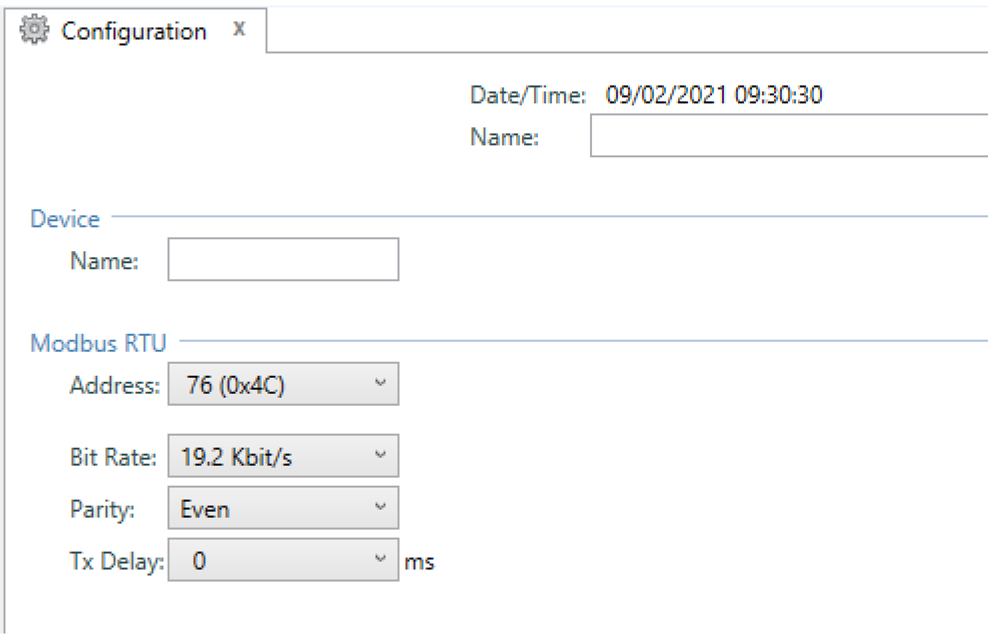
Through the *Update* link it is possible to update the device firmware.

Through the *Reset* link it is possible to reset the device.



5.2 Configuration

In Omni Automation IDE, in the tree view list on the left, double-clicking on *Configuration*, visible under the drive, it opens the configuration tab of the Modbus RTU communication.



When a value in the configuration is modified, a box appears to highlight that the new value entered differs from the value stored in the device. To transfer and store the new value in the device it is necessary to press the *Write* button.

The *Date/Time* field shows the date and the time when the configuration was last written. The next field, *Name*, allows instead to assign a name to the configuration.

Within the *Device* section it is possible to assign a name to the device.

The *Modbus RTU* section contains the following fields:

Address, it allows to assign the device address that must be unique within the mains line.

Bit Rate, it allows to select the communication speed.

Parity, it allows to choose whether the transmitted and received characters have the parity bit or not and, in case of parity, to select the type.

Tx Delay, it is useful for delaying the response of the server in order to allow the client to switch the receiving line.

Changes to the *Modbus RTU* section are active at the next restart of the device.

5.3 Status and Diagnostics

In Omni Automation IDE, on the left in the tree view list of the connected devices, double-clicking on the voice *Status* visible under the drive, opens the tab that shows the status of the device and the eventual errors.

The check box *Periodic Update*, when selected, maintains updated the device status display. By removing the check mark from the box, to update the status you will need to click on the near link *Update*.

The link *Update* is activated when the check box is not selected and allows you to manually update the device status display.

In the *Device* section, the field *Powered* indicates the time elapsed since the device was turned on or since the last reset.

Under the section *Device,* the field *Vp* shows the voltage value of the DC power bus inside the drive.

Under the section *Device,* the field *Temperature* shows the power stage temperature value of the drive.

Under the section *Digital I/O,* it is shown the status of the digital inputs and outputs. When the signal is associated with the yellow color it means that it is in the Active status while if the color is grey it is in the Inactive status.

In the *Analog I/O* section, the fields *Input* 0 and *Input 1* show the voltage value applied to the corresponding inputs. Similarly, the fields *Output 0* and *Output 1* indicate the voltage value present at the corresponding analog outputs.

Following is the description of the fields inside the section *Motor*.

The field *Position* shows the actual position reached by the motor. The integer part of the value indicates the number of complete revolution, while the decimal part shows the fraction of the revolution reached by the motor with a resolution of 1/10000 rev. For example, the value 0.5000 indicates that the motor is half revolution forward with respect to 0 position, while the value -3.7500 indicates that the motor is 3 and 3 quarters of revolution backward with respect to the 0 position.

The field *Speed* shows the actual speed reached by the motor.

The field *Current* shows the actual phase current which flows in the motor phases. It is not surprising if the field value differs from the configuration because, particularly at high speed, or with low supply voltage, due to the inductance and counter-electromotive force of the motor, the current cannot reach the set rated value.

The field *Load Ratio* becomes visible only when the drive operates at closed-loop and indicates the relation between the torque supplied by the motor and the load resistant torque. The value is expressed in percentage and when it reached 100% the motor stops. This condition, in fact, indicates that the load resistant torque has exceeded the one supplied by the motor. If the value is positive it means that the load applies a resisting torque in the direction opposite to that of the motor rotation, while if the value is negative it means that the load is trying to drag the motor, in the same direction of the rotation, beyond its position.

Under the section *Motor Encoder,* the field *Value* indicates the cyclical position of the encoder over a revolution. The value, expressed in 4x encoder resolution, resets to zero at each revolution of the encoder itself.

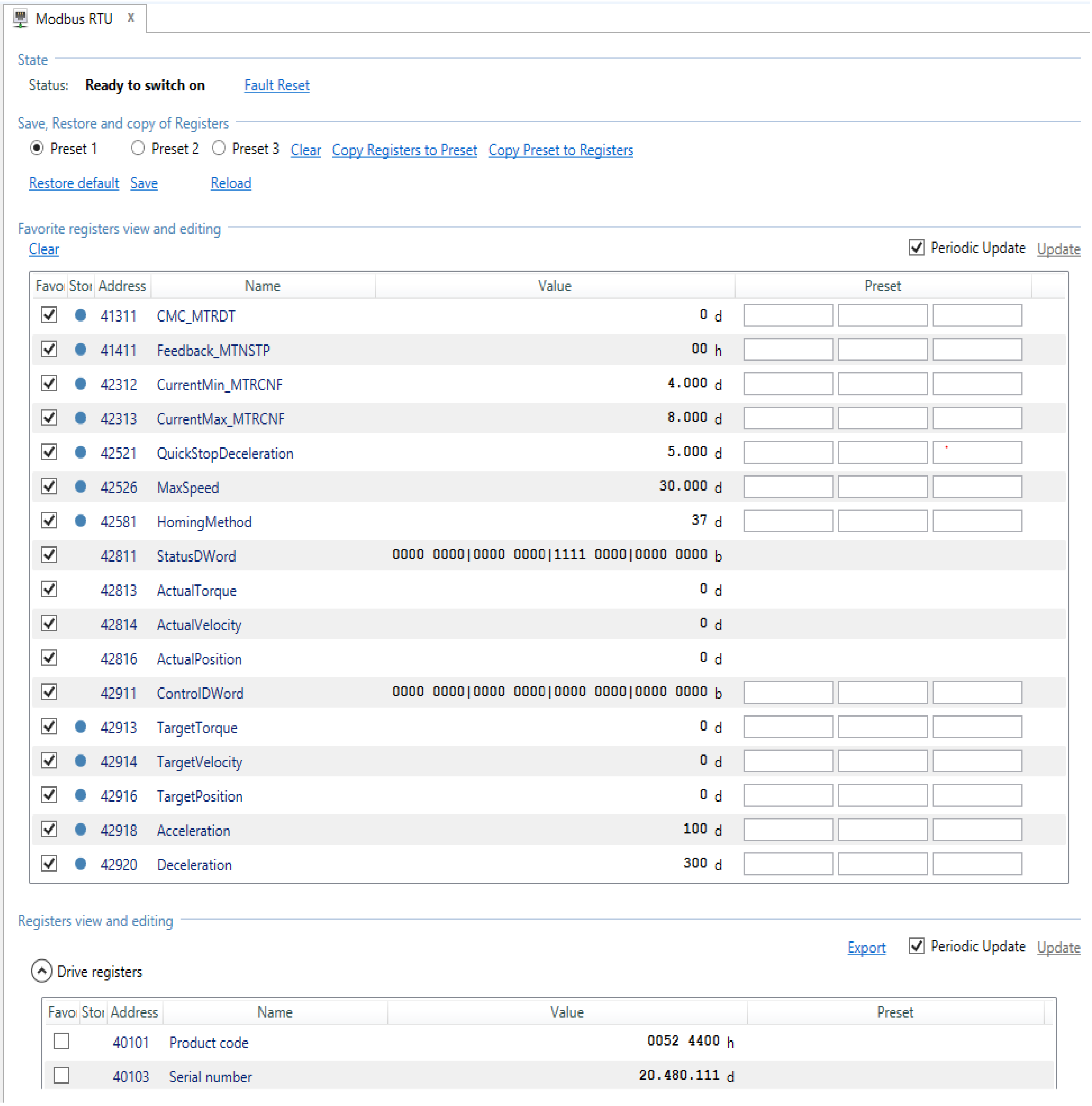
Under the section *Motor Encoder,* the field *Frequency* indicates the frequency of the encoder's A and B signals.

Under the section *Motor Encoder,* it is possible to know the logic level of the *Phases* A, B and Z of the encoder according to the associated color. When the signal is associated with the yellow color it means that it is in the Active status, while if the color is grey it means that the signal is in the Inactive status.

The section *Errors/Fault*, in table form, shows the history of the errors occurred from the last power on. When the error is active, the column *Active* of the table contains an exclamation mark, while when the error is stored but no more active the column *Active* is empty and the line background is red. When the error reset is executed the background becomes white. In the table are stored up to the latest 10 errors, then the latest replace the oldest. The column *Time* shows the moment when the error occurred after the power on of the drive. The column *Code* contains the error numeric code while the column *Details* shows a brief description of the error. Positioning the mouse pointer over the contents of each column a tooltip provides more details.

5.4 Visualization and modification of registers

In Omni Automation IDE, on the left in the tree view list of the connected devices, double-clicking on the entry *Modbus RTU*, visible below the drive, you access the tab that allows to view and edit the device registers.



Scrolling the tab from top to bottom the first section met shows the device status. The link *Fault Reset* on the right allows you to reset faults and errors that are no longer active.

Removing the fault, the drive can immediately go to the operating state and move the motor according to the contents of the registers. Before removing the fault, properly configure the registers and make sure that an unexpected movement of the motor cannot cause any damage to things or people.

The next selection allows to select the Preset group in which to transfer the register values or the Preset group to be used to write the registers themselves.

The Presets are locations containing a value that can then be transferred to a corresponding register.

To transfer the Preset value into the register, simply press the Enter button or double-click with the left mouse button.

In total. 3 Presets are available for each register that allow you to quickly switch from a value to another simply by double-clicking in the field containing the desired value.

The link *Save* allows you to save the contents of the registers in the non-volatile memory while *Restore default* cancels the contents of the non-volatile memory restoring the default value of the registers. Both operations are possible only with the motor disabled. For more details see chapter 6.2 Saving and restoring of default values.

The registers visualization and modification area is logically divided in two; in the upper area there are the most common use registers (favorite) while in the lower one there are all the drive registers ordered by address.

The *Periodic Update* check box, when selected, maintains updated the visualization of the device status. By removing the check from the box, to update the status it will be necessary to click on the near link *Update*.

The link *Update* is activated when the check box is not selected and allows you to manually update the visualization of the drive status.

The registers that you want to monitor frequently can be added to the favorites area by checking the box in the *Favorite* column of the table.

The registers contained in the favorites area are more frequently updated than the others, allowing a more accurate analysis of the value.

The column *Storable* of the table indicates whether the register can be saved in the non-volatile memory using the *Store Parameters* functions. If there is a blue dot it means that the register will be stored in the non-volatile memory as a result of the *Store Parameters* command.

The following columns show the register address and the name.

The *Value* column shows the register value in real time.

Every time the value changes, the background becomes green for about 5 seconds to highlight the change of the value.

The registers value can be visualized in decimal, hexadecimal or binary format. To change the visualization base, simply place the mouse over the value and click with the right button, then select the desired visualization base.

The letter after the value indicates the base in which the number is visualized according to the following correspondence:

Symbol	Base
d	Decimal
h	Hexadecimal
b	Binary

In the *Preset* column there are three fields that can be filled with the values you want to write in the corresponding register. By pressing the enter button or double-clicking with the left mouse button, the value of the field is copied into the register.

Please note that some registers prevent writing in specific operative conditions. For example, it is not possible to modify the motor parameters (as *Inductance_MTRDT*) when the drive is enabled.

The Preset can be compiled with a hexadecimal value, putting the prefix “0x”, or binary, putting the prefix “0b”.

For clarity of writing, spaces can be inserted to logically separate figures. For example, it is possible to write the value 212 in binary as “0b 1101 0100 to highlight more clearly the nibbles (groups of 4 bits) which compose the byte.

The link *Export*, in the registers area, allows you to export the contents of each dictionary register to a file, in CSV format.

By clicking with the right button of the mouse on the *Modbus RTU* item (on the left in the devices tree) it is possible to save and load the configuration of the *Modbus RTU* tab which includes the registers in the favorites area, the value display base, etc..

6 Operation

The drive operates mainly in slave mode through the *Modbus RTU*. With an appropriate configuration it is also possible to control some functions through the I/O signals integrated in the device (for example the alarms reset).

ATTENTION, verify carefully that there are no conflicts between the control via fieldbus and the local control, in order to prevent unexpected movements or a failure in the activation of desired functions.

The operating modes supported by the firmware revision described in this manual are 4, as shown in the table below:

Abbreviation	Description
pp	Position mode
pv	Velocity mode
tq	Torque mode
hm	Homing mode

Note: It is suggested to always verify if there are firmware upgrades with new operating modes or new implemented functionalities.

Before operating, the drive requires some preliminary settings by writing appropriate value in the registers that compose the dictionary of the device itself.

6.1 Minimum settings

The minimum settings that must be made before enabling the motor involves the setting of the motor parameters (registers from 41311 to 41323) and the setting of the running and idle current (42312 *CurrentMin_MTRCNF* and 42313 *CurrentMax_MTRCNF* registers).

6.1.1 Motor parameters setting

It is very important to set the motor parameters correctly to achieve smooth movement, best dynamic performance and best efficiency.

It follow a detailed description of the registers involved in the motor settings.

6.1.1.1 CMC_MTRDT

If you are using a LAM Technologies motor just compile the *CMC_MTRDT* register with the CMC motor and automatically the drive will use the optimal configuration for the chosen motor.

The following table shows the correspondence between the CMC code and the motors. For the motors which allow a *Bipolar Parallel* or *Bipolar Series* phase connection, different CMC codes are shown, as they vary in the electrical features and dynamics.

LAM Technologies motors CMC code

CMC	Motor	Typo of connection
	NEMA 17	
130200	M1173020	Unchangeable
130210	M1173021	Unchangeable
130300	M1173030	Unchangeable
130310	M1173031	Unchangeable
130400	M1173040	Unchangeable
130410	M1173041	Unchangeable
130510	M1173051	Unchangeable
	NEMA 23	
230110	M1233011	Unchangeable
230120	M1233012	Unchangeable
230210	M1233021	Unchangeable
230220	M1233022	Unchangeable
230310	M1233031	Unchangeable
230320	M1233032	Unchangeable
230410	M1233041	Unchangeable
230510	M1233051	Unchangeable
230610	M1233061	Unchangeable
230620	M1233062	Unchangeable
230640	M1233064	Unchangeable
230700	M1233070	Unchangeable
230710	M1233071	Unchangeable
	NEMA 24	
530410	M1243041	Unchangeable
530420	M1243042	Unchangeable
530440	M1243044	Unchangeable
	NEMA 34	
330110	M1343011	Bipolar Parallel
330111	M1343011	Bipolar Series
330200	M1343020	Bipolar Parallel
330201	M1343020	Bipolar Series
330210	M1343021	Bipolar Parallel
330211	M1343021	Bipolar Series
330310	M1343031	Bipolar Parallel
330311	M1343031	Bipolar Series
330410	M1343041	Bipolar Parallel
330411	M1343041	Bipolar Series
330500	M1343050	Bipolar Parallel
330501	M1343050	Bipolar Series
330510	M1343051	Bipolar Parallel
330511	M1343051	Bipolar Series

330600	M1343060	Bipolar Parallel
330601	M1343060	Bipolar Series
	NEMA 42	
430100	M1433010	Bipolar Parallel
430101	M1433010	Bipolar Series
430200	M1433020	Bipolar Parallel
430201	M1433020	Bipolar Series
430400	M1433040	Bipolar Parallel
430401	M1433040	Bipolar Series
	NEMA23 with Encoder	
230410101	M1241E106	Unchangeable
230620101	M1262E106	Unchangeable
230640101	M1264E106	Unchangeable
230700101	M1270E106	Unchangeable
230710101	M1271E106	Unchangeable
	NEMA34 with Encoder	
330200101	M1320E106	Unchangeable
330210101	M1321E106	Unchangeable
330201101	M1325E106	Unchangeable
330211101	M1326E106	Unchangeable
330310101	M1331E106	Unchangeable
330311101	M1336E106	Unchangeable
330500101	M1350E106	Unchangeable
330501101	M1355E106	Unchangeable
330600101	M1360E106	Unchangeable
330601101	M1365E106	Unchangeable

If the motor is not listed in the table, it may have been introduced recently. Usually the CMC code is shown in the datasheet of the motor and in the dedicated page on the website. If you have difficulty in finding it you can request it by writing to support@lamtechnologies.com.

6.1.1.2 PolePairs_MTRDT

The *PolePairs_MTRDT* register allows to set the number of motor poles.

The drive uses this information to properly relate the internal position with the one of the motor.

In case of a two-phases stepper motor each pole gives rise to 4 full steps, therefore a motor of 200 steps/rev (1.8° step angle) requires to set a value equal to 50 (200 / 4). If, for example, your motor has 100 steps/rev, you will set the value 25 or the value 100 if you are using a motor of 400 steps/rev. (0.9° steps angle.

6.1.1.3 Resistance_MTRDT

The *Resistance_MTRDT* register must be compiled with the correct value of the motor phase resistance. Each unit is worth 10mOhm (i.e. 0.01Ohm) then, for example, to set a value of 3.5Ohm it is necessary to write the value 350 (3.5 / 0.01) in the parameter.

The motor phase resistance is normally specified by the manufacturer and shown in the motor technical datasheet.

Some motors allow more types of phases connection and in this case it is necessary to verify for which connection is specified the resistance value and adapt it to the phase connection chosen to connect the motor to the drive. The following table shows the conversion factors to be used:

Phase connection with which the Manufacturer has characterized the resistance	Connection chosen for the phases		
	Unipolar	Bipolar Parallel	Bipolar Series
Unipolar	Not supported	0.5	2
Bipolar Parallel	Not supported	1	4
Bipolar Series	Not supported	0.25	1

For example, if the motor has a characteristic resistance of 2.2Ohm in unipolar and is connected to the drive with the phases set in bipolar parallel, the *Resistance_MTRDT* register will have to be compiled with the value 110 (2.2 * 0.5 / 0.01); instead, in case of a bipolar series connection the value to be inserted in the *Resistance_MTRDT* register will be 440 (2.2 * 2 / 0.01).

If a two-phase motor has 4 wires, it means that the type of phase connection has been already decided during production and the resistance value specified by the manufacturer is therefore the one to be written in the *Resistance_MTRDT* , register without any further processing.

In the event that the value of the resistance is unknown, it is possible to measure it through an ohmmeter. It is suggested to carry out the measurement with the phases already connected in the chosen configuration, furthermore it is a good idea to average the value through repeated measurements on several motors, if available.

6.1.1.4 Inductance_MTRDT

The *Inductance_MTRDT* register must be filled in with the correct value of the motor phase inductance. Each unit is worth 10uH (i.e. 0.01mH) therefore to set, for example, a value of 4.2mH it is necessary to write the value 420 (4.2 / 0.01) in the register.

Some motors allow more types of phase connection and in this case it is necessary to verify for which connection the inductance value is specified and adapt it to the phase connection chosen to connect the motor to the drive. The following table shows the conversion factors to use:

Phase connection with which the Manufacturer has characterized the inductance	Connection chosen for the phases		
	Unipolar	Bipolar Parallel	Bipolar Series
Unipolar	Not supported	1	4
Bipolar Parallel	Not supported	1	4
Bipolar Series	Not supported	0.25	1

For example, if the motor has a characteristics inductance of 1.6mH in unipolar and is connected to the drive with the phases in bipolar parallel, the *Inductance_MTRDT* register must be filled in with the value 160 (1.6 * 1 / 0.01); instead, if you choose a bipolar series connection the value to be entered will be of 640 (1.6 * 4 / 0.01).

If a two-phase motor has 4 wires it means that the type of phase connection has already been decided during production and the inductance value specified by the manufacturer is therefore the one to be used for the *Inductance_MTRDT* register, without any further processing.

In the event that the value of the motor inductance is not known, it can be measured through an inductance meter. We suggest you to perform the measurement with the phases already connected in the configuration chosen for the drive, and it is also good practice to average the value through repeated measurements on different motors, if available.

6.1.1.5 *BackEMF_MTRDT*

The *BackEMF_MTRDT* register must be filled in with the value of the counter-electromotive force generated by the motor at a speed of 1000rpm. Each unit is worth 10mV (i.e. 0.01V) therefore, for example, if the motor generates 25V at 1000rpm it is necessary to write the value 2500 (25 / 0.01) in the *BackEMF_MTRDT* register.

The counter-electromotive force is normally specified by the manufacturer in the motor technical datasheet.

Some motors allow more types of phase connection and in this case it is necessary to verify for which connection the value of counter-electromotive force is specified and adapt it to the phase connection chosen to connect the motor to the drive. The following table shows the conversion factors to use:

Phase connection with which the Manufacturer has characterized the counter-electromotive force	Connection chosen for the phases		
	Unipolar	Bipolar Parallel	Bipolar Series
Unipolar	Not supported	1	2
Bipolar Parallel	Not supported	1	2
Bipolar Series	Not supported	0.5	1

For example, if the motor has a counter-electromotive force of 28V at 1000rpm in unipolar and is connected to the drive with the phases in bipolar parallel, the *BackEMF_MTRDT* register must be filled in with the value 2800 (28 * 1 / 0.01); instead, if you choose a bipolar series connection the value to be entered will be of 5600 (28 * 2 / 0.01).

If a two-phase motor has 4 wires it means that the type of phase connection has already been decided during production and the value of the counter-electromotive force generated by the motor is therefore the one to be used for the *BackEMF_MTRDT* register, without any further processing.

In the event that the value of the counter-electromotive force is not known, it can be measured through an AC voltmeter connected to one phase of the motor and make it rotate at such a speed to produce a BEMF with a frequency of about 50Hz. Successively you need to relate the measured voltage value to the speed of 1000rpm using the following formula:

Vbemf = (Vac * 1000) / Mrpm

Where Vbemf is the value of the counter-electromotive force expressed in V, Vac is the measured voltage expressed in V and Mrpm is the speed at which the motor was made to rotate expressed in RPM. For example, if the motor has been rotated at a speed of 60rpm (to obtain 50Hz) and the measured voltage value was of 4.85Vac, the Vbemf value will be equal to 80.83 (4.85 * 1000 / 60) and the value to be entered in the *BackEMF_MTRDT* register will be (80.83 / 0.01).

Note that the frequency of the BEMF is related to the motor speed through the number of poles, according to the following relationship:

Fhz = Npl * Mrpm / 60

Where Fhz is the frequency of the BEMF expressed in Hz, Npl is the number of motor poles (non-dimensional) and Mrpm is the rotation speed of the motor. For example, if we rotate a 50 poles motor (corresponding to a step angle of 1.8°) at 100rpm, we obtain a BEMF frequency equal to about 83.3Hz.

It is suggested to repeat the measurement on several motors and average the results obtained.

It is essential that the value entered in the *BackEMF_MTRDT* register corresponds to the counter-electromotive force generated at a speed of 1000rpm.

6.1.1.6 *RatedCurrent_MTRDT*

The *RatedCurrent_MTRDT* register must be filled in with the motor rated current. Each unit is worth 10mArms (i.e. 0.01 Arms) then to set, for example, a value of 4.2Arms it is necessary to write the value 420 (4.2 / 0.01) in the register.

The value written in the *RatedCurrent_MTRDT* register must take into account the phases connection chosen for the motors that provide more possibilities. According to the connection chosen for the phases and to the configuration chosen by the manufacturer to characterize the rated current, it is necessary to consider one of the scaling factors shown in the table below:

Phase connection with which the Manufacturer has characterized the current	Connection chosen for the phases		
	Unipolar	Bipolar Parallel	Bipolar Series
Unipolar	Not supported	1.41	0.707
Bipolar Parallel	Not supported	1	0.5
Bipolar Series	Not supported	2	1

For example, if the manufacturer specifies a current of 2A for a bipolar parallel connection and the motor is connected to the drive in bipolar parallel, no conversion is needed and the *RatedCurrent_MTRDT* register can be set with the value 200 (2 / 0.01). Instead, if a bipolar series connection is chosen, the register must be filled in with the value (2 * 0.5 / 0.01).

If a two-phase motor has 4 wires it means that the type of phase connection has already been decided during production and the value of the rated current specified by the manufacturer is therefore the one to be entered in the field *Current,* without any further processing.

It is essential that the value in the *RatedCurrent_MTRDT* register corresponds exactly to the rated current of the motor and that this parameter is never used to set the working current. The working current of the motor is set through the *CurrentMax_MTRCNF* register described further on.

6.1.1.7 MaxCurrent_MTRDT

The *MaxCurrent_MTRDT* register must be filled in with the maximum current to which the motor can be supplied. Each unit is worth 10mArms (i.e. 0.01Arms) then , for example, to set a value of 5.0Arms it is necessary to write the value 500 (5.0 / 0.01) in the parameter.

In the event that the data is not available in the motor datasheet, it is suggested to use the same value of the 4318_h *RatedCurrent_MTRDT* register.

The value written in the *MaxCurrent_MTRDT* register must consider the connection chosen for the phases for the motors which provide more possibilities. See previous chapter 6.1.1.6 RatedCurrent_MTRDT.

6.1.1.8 RatedTorque_MTRDT

The *RatedTorque_MTRDT* register must be filled in with the static torque value of the motor when supplied at the rated current. Each unit is worth 10mNm (i.e. 0.01Nm) then , for example, to set a value of 6.8Nm it is necessary to write the value 6800 (6.8 / 0.01) in the parameter.

The static torque value is normally specified by the manufacturer in the motor datasheet and is often called *Holding Torque*. If the value is expressed in a unit of measurement other than Nm, it can be converted using the coefficients shown in the table below:

	Newton Centimeter (N-cm)	Newton Meter (N-m)	Pound Force Inch, (lbf-in)	Ounce Force Inch, (ozf-in)
Newton Centimeter (N-cm)	**1 N-cm**	**0.01 N-m**	**0.0885 lbf-in**	**1.42 ozf-in**
Newton Meter (N-m)	**100 N-cm**	**1 N-m**	**8.85 lbf-in**	**142 ozf-in**
Pound Force Inch, (lbf-in)	**11.3 N-cm**	**0.113 N-m**	**1 lbf-in**	**16 ozf-in**
Ounce Force Inch, (ozf-in)	**0.706 N-cm**	**0.00706 N-m**	**0.0625 lbf-in**	**1 ozf-in**

Some motors allow more types of phase connection and in this case it is necessary to verify for which connection the rated torque is specified and adapt it to the phase connection chosen to connect the motor to the drive. The following table shows the conversion factors to use:

Phase connection with which the Manufacturer has characterized the rated static torque	Connection chosen for the phases		
	Unipolar	Unipolar	Unipolar
Unipolar	Not supported	1.41	1.41
Bipolar Parallel	Not supported	1	1
Bipolar Series	Not supported	1	1

For example, if a motor has a rated static torque of 3.1Nm in unipolar and is connected to the drive with the phases in bipolar parallel or in bipolar series, the *RatedTorque_MTRDT* register must be filled in with the value 437 (3.1 * 1.41 / 0.01).

If a two-phase motor has 4 wires it means that the type of phase connection has already been decided during production and the *Holding Torque* value specified by the manufacturer is therefore the one to be used, without any further processing.

In the event that the value of the motor torque is not known, it can be measured through a torquemeter with the motor phases supplied at the rated current. It is suggested to perform the measurement with the phases already connected in the configuration chosen for the drive, furthermore it is a good practice to average the value through repeated measurements on several motors, if available.

It is essential <u>that the value in the *RatedTorque_MTRDT* register corresponds exactly to the rated torque of the motor.</u>

6.1.1.9 *MaxSpeed_MTRDT*

The *MaxSpeed_MTRDT* register must be filled in with the maximum speed that the motor can reach. Each unit is worth 0.1rpm therefore, for example, to set a value of 600rpm it is necessary to write the value 6000 (600 / 0.1) in the register.

6.1.2 Running and idle current configuration

The drive allows to freely define the running and idle current of the motor to optimally adapt it to the application.

The running current is impressed on the motor during the rotation while the idle current is applied to the motor after the stop. The time from the motor stop, after which the current is set to the idle value, is configurable.

When the motor is equipped with Encoder and the drive configured for closed-loop control, it is also possible to set the current regulation so that it dynamically adapts to the load applied to the motor ($43A1_h$ *Mode_CRRG* register).

When the current regulation is configured dynamically, the running and idle current correspond respectively to the current supplied to the motor in absence of load and to the current at full load (locked rotor).

Below is a description of the parameters relating to the motor operating current.

6.1.2.1 CurrentMin_MTRCNF

The *CurrentMin_MTRCNF* register allows to specify the current applied to the motor in idle mode. The current is applied after the motor stop after the time defined by the *CurrentIdleDelay_MTRCNF* register described below.

The register is expressed as a percentage of the motor rated current (4318_h *RatedCurrent_MTRDT* register) and each unit is equal to 0.01%. For example, if you want to set an idle current equal to the 30% of the rated current, it is necessary to write the *CurrentMin_MTRCNF* register with the value 3000 (30 / 0.01). If the configured motor rated current is for example 4Arms, the idle current will be equal to 1.2Arms (30% of 4A).

6.1.2.2 CurrentMax_MTRCNF

The *CurrentMax_MTRCNF* register allows to specify the current applied to the motor during rotation.

The register is expressed as a percentage of the motor rated current (4318_h *RatedCurrent_MTRDT* register) and each unit is equal to 0.01%. For example, if you want to set a running current equal to the 80% of the rated current, it is necessary to write the *CurrentMax_MTRCNF* register with the value 8000 (80 / 0,01). If the configured motor rated current is for example 4Arms, the idle current will be equal to 3.2Arms (80% of 4A).

6.1.2.3 CurrentIdleDelay_MTRCNF

The *CurrentIdleDelay_MTRCNF* register allows to specify the waiting time from the motor stop before the current is set to the value defined by the *CurrentMin_MTRCNF* register.

Each unit is equal to 1ms, therefore setting for example the value 500, the drive will wait for 500ms from the motor stop before changing the phase current.

6.2 Saving and restoring of default values

The device is able to save many of its registers in the non-volatile memory. The registers that can be saved are highlighted with the symbol 🖫 in the field *Note* of the table that describe the register itself, as in the following example:

Name	Mnemonic
PDO Mapping	Note 🖫
Maximum	Unit
	Descrizione

When the value of a register is saved in the non-volatile memory it is automatically restored at the power on or in case of reset.

By saving a value other than the default it is possible to adapt the device to the application without having to configure it every time. Apparently this seems to be a simplification but it forces to prepare the device (saving the desired values in the registers) before it can be used in the application. When there are many applications, or they are updated over time, you are forced to keep an archive with all the registers values used in each application and in each version and over time this can become complex and give rise to errors. On the contrary, by delegating to the master controller the configuration of each drive, it will not be necessary to keep track of how the device should be initialized. In this case, in fact, the specific application will initialize the registers with the desired values and without errors. Furthermore, if the device should need to be replaced, the technical support can simply send a new device without worrying about the application and the version in which it will be installed.

When possible it is therefore recommended not to use the *Save* function to modify the default value of the registers. On the contrary, it is recommended to always initialize every register used in the application with the desired value, independently from the saving or default. The initialization must be repeated in case of device reset.

Saving is done by writing an appropriate key in the *SaveAllParameters_SPF* register or using the Omni Automation IDE software.

The registers can be also be restored to the default value writing an appropriate key in the *RestoreAllDefaultParameters_RDP* or using the Omni Automation IDE software.

The defaults values can be saved or restored only with the motor disabled. Trying the operation with the motor enabled an error code is received.

It is possible to save or restore the default values for 10,000 times max.

6.3 Motor Holding Brakes

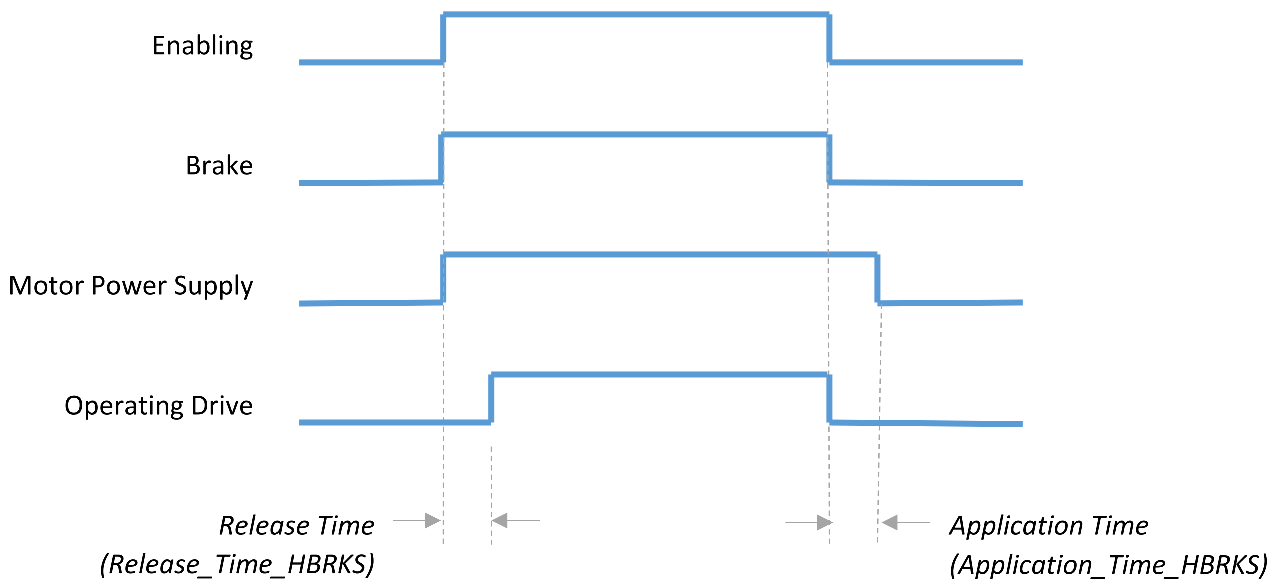
The drive is able to control the holding brake of the motor through one of the digital outputs.

Through the *Option_HBRKS* register it is possible to set the drive to handle automatically the brake with motor enabling, taking into account the brake's characteristic engaging and disengaging time.

With the brake control enabled, by enabling the motor, the drive activates immediately the digital output predisposed to control the break to release it and to simultaneously supply the motor to keep it in position. The transition to the operating state is delayed by the time required to the brake to completely disengage. This time can be freely set through the *Release_Time_HBRKS* register.

With the brake handling enabled, by disabling the motor, the drive immediately exits the operating state and simultaneously deactivates the digital output predisposed to the control of the brake to engage it. The motor remains powered for the time the brake requires to completely engage. This time can be freely set through the *Application_Time_HBRKS* register.

The following graph shows the temporal relationship between the described events:



The *Option_HBRKS* register is also useful to enable and set the manual control of the brake that can intervene in an exclusive way, with respect to the control operated by the drive, or shared.

The manual control can be performed via a digital input configurable through the *Holding_Brake_DIA* register or through the bit0 of the *Control_HBRKC* register.

Through the Status_HBRKC register it is possible to know in real time the status of the brake and the status of the associated output.

When the digital output predisposed to control the brake is active, the brake is considered released.

6.4 SSI Encoder

The drive is able to read rotary and linear encoders with synchronous serial interface (SSI - Synchronous Serial Interface).

Both mono-turn and multi-turn binary encoders can be connected and the total number of bits of the frame, of the multi-turn part and of the single-turn part can be freely set independently. It is also possible to indicate the position of the most significant bit (MSB) inside the frame to correctly read encoders with data center alignment (tree structure) or aligned to the right or left of the frame.

The SSI encoder can be used to initialize the position of the motor (at the power on or on request) and to verify in real time the positioning through the Following Error.

However, when using the SSI encoder it is still possible to continue and use the incremental encoder at the same time, for example for a redundant position control or to use the closed-loop control mode.

The SSI encoder can be mounted on the motor shaft or in a different kinematic position, even after a reduction ratio. Using the *PositionOffset_ENCSSI*, *ScalingFactor_N_ENCSSI* and *ScalingFactor_D_ENCSSI* registers it is possible to bring back the quote read by the SSI encoder to the motor position declared in 0.0001rev. The calculation made by the drive is as follows

Position_ENCSSIV = (*PositionRaw_ENCSSIV* - *PositionOffset_ENCSSI*) * *ScalingFactor_N_ENCSSI / ScalingFactor_D_ENCSSI*

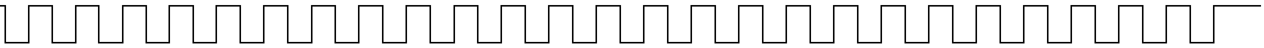
The SSI encoder can be supplied externally or through the +5Vdc output provided on the drive for this purpose.

The drive is able to detect error conditions such as the cut or short-circuit of the connecting cable with the SSI encoder. The error control can be activated by setting bit1 of the *Configuration_ENCSSI* register.

The drive is able to read a wide variety of SSI encoders thanks to the possibility to freely configure the data frame structure. For example, it is possible to use the SSI encoders with the data organized in the center (tree structure), aligned to the left or to the right.

Through the *FrameLength_ENCSSI*, *MSB_Offset_ENCSSI*, *TurnsBits_ENCSSI* and *SingleTurnBits_ENCSSI* registers it is possible to configure respectively the total length of the frame, the position of the most significant bit (MSB), the number of bits used for the revolutions counting and the number of bits used for the single-turn position.

The following example shows an SSI encoder with a 25bits length frame with the data aligned at the center and composed of 8bits for the full revolutions counting and 12bits for the single-turn position measurement.

clock


data	b24	b23	b22	b21	b20	b19	b18	b17	b16	b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
	0	0	0	0	n7	n6	n5	n4	n3	n2	n1	n0	p12	p11	p9	p8	p7	p6	p5	p4	p3	p2	p1	p0	0

b.. Number of bits in the frame

n.. Bit for full revolutions counting

p.. Bit for the single-turn position measurement

To correctly read an SSI encoder with the data frame organized as shown above, the following registers must be configured:

FrameLength_ENCSSI = 25

TurnsBits_ENCSSI = 8

SingleTurnBits_ENCSSI = 12

MSB_Offset_ENCSSI = 4

Setting the bit2 of the *Configuration_ENCSSI* register to 1, the position, expressed in 0,0001rev read by the SSI encoder, is copied into the *Position_actual_value* register at each transition from 0 to 1 of the bit0 of the *Status_ENCSSIV* register and at each transition in the *Operational* state of the drive.

The *Position_actual_value* register can be initialized with the position expressed in in 0,0001rev read by the SSI encoder also through the *Homing* operative mode setting the *Homing_method* register = -80.

For example, if you want to use the SSI encoder to monitor the following error (*Following_error_actual_value*) you need to configure the *Feedback_MTNSTP* register = 0x12;

The *Data* signal of the SSI encoder and the *I (Index)* signal of the incremental encoder share the same connection, therefore when using the SSI encoder the Index signal of the incremental encoder cannot be used and the bit0 of the *Configuration_ENCMTR* register must be set to 0.

6.5 Closed-loop

The drives belonging to the DDS5 series are able to control the closed-loop motor with the following benefits:

- No loss of step
- 100% use of motor torque
- Possibility of control of the motor torque
- Dynamic current supply proportional to the load
- Less motor heating
- Position Error monitoring and Following Error alarm

To perform the closed-loop control the drive needs to know the motor position in real time, therefore the motor must be equipped with an Encoder. This latter must ensure a delay of less than 5us between the real motor position and the one acquired at the sampling point, otherwise the maximum speed allowed by the closed-loop control is reduced.

In order for the drive to operate properly in closed-loop, it is necessary to accurately configure the characteristics of the motor, the encoder and finally enable the closed-loop control mode by setting the bits 1..0 of the *Feedback_MTNSTP* register to 1.

6.6 Use and functionality of the StatusDWord and ControlDWord registers

The *StatusDWord* and *ControlDWord* registers are used by all the operating mode and allow to know the status of the motor and to check its functioning.

The *StatusDWord* register has a dimension of 32bit (double word) and it is read-only. The following table shows the meaning of the bits contained in the register:

<table>
<tr><th colspan="4">StatusDWord</th></tr>
<tr><th>bit</th><th>Description</th><th>bit</th><th>Description</th></tr>
<tr><td>31</td><td>Reserved, ignore the value</td><td>15</td><td>PositionReached, 1=true</td></tr>
<tr><td>30</td><td>Reserved, ignore the value</td><td>14</td><td>VelocityReached, 1=true</td></tr>
<tr><td>29</td><td>Reserved, ignore the value</td><td>13</td><td>TorqueReached, 1=true</td></tr>
<tr><td>28</td><td>Reserved, ignore the value</td><td>12</td><td>Standstill, 1=true</td></tr>
<tr><td>27</td><td>Reserved, ignore the value</td><td>11</td><td>Reserved, ignore the value</td></tr>
<tr><td>26</td><td rowspan="3">OperationMode
<table><tr><th>bit 26..24</th><th>Operation mode</th></tr><tr><td>001</td><td>Position</td></tr><tr><td>010</td><td>Velocity</td></tr><tr><td>011</td><td>Torque</td></tr><tr><td>111</td><td>Homing</td></tr></table>
The bit combinations not in the table must be considered as non-selected operation mode.</td><td>10</td><td>ReferenceSet, 1=true</td></tr>
<tr><td>25</td><td>9</td><td>TriggerAcknowledge</td></tr>
<tr><td>24</td><td>8</td><td>Busy, 1=operation in progress</td></tr>
<tr><td>23</td><td>Reserved, ignore the value</td><td>7</td><td>Reserved, ignore the value</td></tr>
<tr><td>22</td><td>Reserved, ignore the value</td><td>6</td><td>Reserved, ignore the value</td></tr>
<tr><td>21</td><td>Reserved, ignore the value</td><td>5</td><td>Warning, 1= active</td></tr>
<tr><td>20</td><td>Reserved, ignore the value</td><td>4</td><td>Fault, 1=active</td></tr>
<tr><td>19</td><td>PositionError, 1=true</td><td>3</td><td>Halt, 1= active</td></tr>
<tr><td>18</td><td>VelocityError, 1=true</td><td>2</td><td>QuickStop, 1=active</td></tr>
<tr><td>17</td><td>Reserved, ignore the value</td><td>1</td><td rowspan="2">Enable
<table><tr><th>bit 1</th><th>bit 0</th><th>Description</th></tr><tr><td>0</td><td>0</td><td>Disabled</td></tr><tr><td>0</td><td>1</td><td></td></tr><tr><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>1</td><td>Enabled</td></tr></table></td></tr>
<tr><td>16</td><td>Reserved, ignore the value</td><td>0</td></tr>
</table>

Reserved bits must be ignored. Whatever is their value the application must not change its behavior.

The *ControlDWord* register also has a dimension of 32bit (double word). The meaning of the bits is summarized in the following table:

<table>
<tr><th colspan="4">ControlDWord</th></tr>
<tr><th>bit</th><th>Description</th><th>bit</th><th>Description</th></tr>
<tr><td>31</td><td>Reserved, set to 0</td><td>15</td><td>Reserved, set to 0</td></tr>
<tr><td>30</td><td>Reserved, set to 0</td><td>14</td><td>Reserved, set to 0</td></tr>
<tr><td>29</td><td>Reserved, set to 0</td><td>13</td><td>Reserved, set to 0</td></tr>
<tr><td>28</td><td>Reserved, set to 0</td><td>12</td><td>Reserved, set to 0</td></tr>
<tr><td>27</td><td>Reserved, set to 0</td><td>11</td><td>Reserved, set to 0</td></tr>
<tr><td>26</td><td rowspan="3">OperationMode
<table><tr><th>bit 26..24</th><th>Operation mode</th></tr><tr><td>000</td><td>None</td></tr><tr><td>001</td><td>Position</td></tr><tr><td>010</td><td>Velocity</td></tr><tr><td>011</td><td>Torque</td></tr><tr><td>111</td><td>Homing</td></tr></table>
The bit combinations not in the table must be avoided.</td><td>10</td><td>AbsoluteRelative, 1=Relative</td></tr>
<tr><td>25</td><td>9</td><td>Trigger, ↑=trigger</td></tr>
<tr><td>24</td><td>8</td><td>TriggerMode, 1=Active</td></tr>
<tr><td>23</td><td>Reserved, set to 0</td><td>7</td><td>Reserved, set to 0</td></tr>
<tr><td>22</td><td>Reserved, set to 0</td><td>6</td><td>Reserved, set to 0</td></tr>
<tr><td>21</td><td>Reserved, set to 0</td><td>5</td><td>WarningAcknowledge, ↑=accepted</td></tr>
<tr><td>20</td><td>Reserved, set to 0</td><td>4</td><td>FaultAcknowledge, ↑=accepted</td></tr>
<tr><td>19</td><td>PositionErrorLatch, 1=Latch</td><td>3</td><td>Halt, 1= active</td></tr>
<tr><td>18</td><td>VelocityErrorLatch, 1=Latch</td><td>2</td><td>QuickStop, 1=active</td></tr>
<tr><td>17</td><td>Reserved, set to 0</td><td>1</td><td rowspan="2">Enable
<table><tr><th>bit 1</th><th>bit 0</th><th>Description</th></tr><tr><td>0</td><td>0</td><td rowspan="3">Disabled</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr><tr><td>1</td><td>1</td><td>Enabled</td></tr></table></td></tr>
<tr><td>16</td><td>Reserved, set to 0</td><td>0</td></tr>
</table>

The reserved bits must set to 0 to ensure the compatibility with the future firmware releases.

6.6.1 Motor Enabling

The motor is enabled setting the bit1 and bit0 of the *ControlDWord* register to 1. If there are no active fault and the drive conditions allow it (for example the power supply is within the allowed range) the motor is enabled and start to operate in the selected operating mode (for example *Position* if the bit26..24 have value 001).

The enabling of the motor is confirmed by the bit1 and bit0 of the *StatusDWord* which both take value 1.

To disable the motor you just set the bit1 and bit0 of the *ControlDWord* register to 0.

Note that the drive can autonomously disable the motor, for example in case of fault.

6.6.2 Emergency Stop (QuickStop)

By setting the bit2 of the *ControlDWord* register to 1 the emergency stop is activated. Once activated, the emergency stop cannot be interrupted. During the emergency stop the motor decelerates until it stops with the deceleration ramp defined by the *QuickStopDeceleration* register.

The bit2 of the *StatusDWord* register assumes the value 1 each time the drive performs an emergency stop.

By setting the bit2 of the *ControlDWord* register to 0 the emergency stop request is removed and if the procedure is completed (the motor has stopped) the drive returns to operate according to the chosen mode.

Note that the emergency stop can be automatically activated in the event of an emergency such as the activation of a limit switch, etc. If the bit2 of the *StatusDWord* register is set to 1 and the bit2 of the *ControlDWord* register is set to 0, it means that an emergency occurred that required an arrest through QuickStop. In this case the motor is not disabled and the bit4 *Fault* of *StatusDWord* is not activated. To exit the Quickstop condition and return to the operating status, it is necessary to set the bit4 of *ControlDWord* to 1. Note that the drive uses the rising edge of bit4 (transition from 0 to 1) to exit the emergency condition. If bit4 is already set to 1 when the emergency occurs, it is first necessary to set it to 0 and then again to 1 to confirm acceptance of the emergency.

6.6.3 Motor Stop (Halt)

By setting the bit3 of the *ControlDWord* register to 1, the motor stops with the deceleration ramp defined by the *Deceleration* register.

The bit3 can be set to 0 at any time and in this case the motor starts again to move according to the selected operation mode and to the position, velocity and torque targets, accelerating with the acceleration ramp defined by the *Acceleration* register.

The *StatusDWord* register signals the Halt activation by setting the bit3 to 1.

6.6.4 Fault and Warning

When the drive detects a fault condition, it sets the bit4 of *StatusDWord* to 1 and activates appropriate actions such as the emergency stop or the disabling of the motor. The action taken depends on the nature of the fault.

To accept and remove the fault signal it is necessary to set the bit4 of *ControlDWord* to 1. If the fault condition persists, the drive ignores the bit4 and remains in the fault status. Note that the drive uses the rising edge of bit4 (transition from 0 to 1) to exit the fault condition. If the bit4 is already set to 1 when the fault occurs, it must be first set to 0 and then successively to 1 to confirm acceptance of the fault.

Finally, for some types of faults, the device must be switched off and on again.

When the drive detects warning conditions, it signals them by activating the bit5 of *StatusDWord.* The warning event does not immediately compromise the correct operation of the drive that, therefore, remains in the operating state.

To accept the warning signal it is necessary to set the bit5 of *ControlDWord* to 1. If the warning condition persists, the drive ignores bit5. Note that the drive uses the rising edge of the bit (transition from 0 to 1) to cancel the warning signal. If bit4 is already set to 1 when the fault occurs, it must be first set to 0 and then successively to 1 to confirm acceptance of the warning.

6.6.5 Real-time or delayed target process

The drive is able to process the position, velocity and torque targets in real time or in delayed mode, whenever the trigger is activated.

When the bit8 of *ControlDWord* has a value 0, the motion adapts in real time to the target modification therefore, for example, by modifying the value of the *TargetVelocity* register, the motor speed will immediately start to vary to reach the new value.

Instead, by setting the bit8 of *ControlDWord* to 1, the modification of the targets will not produce an immediate change in motion but only after the activation of the trigger signal. For example, setting a new position using the *TargetPosition* register*,* the motor will maintain its own path (or position) until the trigger signal is activated. Only later the new targets will produce the desired effects on motion.

The registers affected by the bit4 of *ControlDWord* are shown in the table:

Registers that be processed in delayed mode
TargetTorque
TargetVelocity
TargetPosition
Acceleration
Deceleration

The trigger signal that commands the application of the targets when the delayed mode is selected consists of the rising edge (transition from 0 to 1) of the bit9 of *ControlDWord*.

The rising edge (transition from 0 to 1) of the bit9 of *StatusDWord* register confirms the application of the targets.

6.6.6 Absolute or relative positioning

When the drive operates in *Position* mode and the delayed targets mode application is active, it is possible to choose whether the position target (*TargetPosition* register) must interpreted as an absolute or relative quota.

When the bit10 of *ControlDWord* is set to 0 the quota is interpreted in absolute value while when the bit0 is set to 1 the contents of the *TargetPosition* register will be used to make a relative movement, with respect to the previous target, at each trigger activation.

Note that if the targets are processed in real time (bit8=0 of *ControlDWord*) it is not possible to use the relative positioning and the bit10 of *ControlDWord* is ignored.

6.6.7 Target reached and motor halted

In the *StatusDWord* register there are 3 bits which indicate that the commanded target has been reached.

The bit13 assumes the value 1 when the difference between the target torque and the one supplied by the motor is, in absolute value, less than the value set by the *WindowTorqueTime* register, for a time longer than the value set by the *WindowTorqueTime* register.

The bit14 assumes the value 1 when the difference between the target velocity and the actual one of the motor is, in absolute value, less than the value set by the *WindowVelocity* register, for a time longer than the value set by the *WindowVelocityTime* register.

The bit15 assumes the value 1 when the difference between the target position and the one assumed by the motor is, in absolute value, less than the value set by the *WindowPosition* register, for a time longer than the value set by the *WindowPositionTime* register.

The bit12 of the status word indicates instead the halted motor condition and it is set to 1 when the actual speed of the motor is, in absolute value, less than the value set by the *WindowZeroSpeed* register. The bit assumes value 0 if the actual motor speed exceeds, in absolute value, the value set by the *WindowZeroSpeed* register, for a time longer than *WindowZeroSpeedTime*.

6.6.8 Following error

In the *StatusDWord* register there are 2 bits that are activated (bit = 1) in case of a following error.

The following error occurs when the speed or the position differ from the value required by the profile generator beyond a fixed value for a defined time.

The bit18 of *StatusDWord* assumes value 1 if the difference between the actual motor speed and the one required by the profile generator is, in absolute value, greater than the value set by the *WindowVelocityError* register, for a time longer than the value set by the *WindowVelocityErrorTimeOut* register.

The bit19 of *StatusDWord* assumes value 1 if the difference between the actual motor position and the one required by the profile generator is, in absolute value, greater than the value set by the *WindowPositionError* register, for a time longer than the value set by the *WindowPositionErrorTimeOut* register.

The bit18 and bit19 of *ControlDWord* allow to store the activation of the speed and position following error respectively. By setting the bit to 1, if the following error is activated the corresponding bit of *StatusDWord* remains at 1 even after the error has ceased. In this case, to set the following error bit to 0 again simply set the corresponding bit18 and bit19 of *ControlDWord* to 0.

6.6.9 Operation mode

The setting of the operation mode is done through the bit26, bit25 and bit24 of the *ControlDWord* register.

The following table shows the relation between the value of the bits and the selected operation mode:

bit 26..24 *ControlDWord*	Selected operation mode
000	None
001	*Position*
010	*Velocity*
011	*Torque*
111	*Homing*

Combinations other than those shown in the table must be avoided.

The drive confirms the active operation mode by the bit26, bit25 and bit24 of *StatusDWord* register.

The following table shows the relation between the value of the bits and the active operation mode:

bit 26..24 *StatusDWord*	Active operation mode
000	None
001	*Position*
010	*Velocity*
011	*Torque*
111	*Homing*

6.7 Operation Modes

The following paragraphs describe the different operation modes. All the examples assume that the drive is started with the default values and successively configured with the minimum setting described in the previous chapters 6.1 Minimum settings.

6.7.1 Position (pp)

In this mode the drive executes a positioning profile by controlling the torque, the velocity and position of the motor. The movement is performed according to the set values of maximum speed, acceleration and deceleration. The master controller can update the target position simply by updating the *TargetPosition* register.

To select the *Position* mode, simply set the bit26..24 of *ControlDWord* to the value 1 (bit26=0, bit25=0 and bit24=1).

The position, velocity and torque targets can be processed in real or delayed time whenever the trigger is activated, as described in chapter 6.6.5 *Real-time or delayed target process*. Furthermore the master controller can command absolute or relative positioning, as described in chapter 6.6.6 *Absolute or relative positioning*.

The conditions of achieved target and halted motor are can be parameterized as well as the limits associated with the following error, as described in chapters 6.6.7 *Target reached and motor halted* and 6.6.8 *Following error*.

Once the parameters have been set, it is possible to use the only *TargetPosition*, *TargetVelocity*, and *TargetTorque* registers to control position, velocity and torque of the motor.

For example, assuming that the motor is in position 0, by setting the registers as shown in the table below, the motor will execute 12 revolutions unitl it reaches the position 120.000 with a full speed of 400rpm, an acceleration equal to 2000rpm/s and a deceleration of 3500rpm/s. The maximum torque is set at the 88,5% of the rated value.

ControlDWord = 0x01000003
TargetTorque = 885
TargetVelocity = 4000
Acceleration = 2000
Deceleration = 3500
TargetPosition = 120000

6.7.2 Velocity (pv)

In this mode the drive controls the motor in speed, always according to the set acceleration and deceleration ramps and to the maximum allowed speed. The master controller can update the velocity target simply by updating the register *TargetVelocity*.

To select the *Velocity* mode, simply set the bit26..24 of *ControlDWord* to the value 2 (bit26=0, bit25=1 and bit24=0).

The velocity and torque targets can be processed in real or delayed time whenever the trigger is activated, as described in chapter 6.6.5 *Real-time or delayed target process*.

The conditions of achieved target and halted motor can be are parameterized as well as the limits associated with the following error, as described in chapters 6.6.7 *Target reached and motor halted* e 6.6.8 *Following error* respectively.

Once the parameters have been set, it is possible to use the only *TargetVelocity* and *TargetTorque* registers to control velocity and torque of the motor.

For example, by setting the process data as shown in the table below, the motor will reach the speed of 640,5rpm, with an acceleration equal to 500rpm/s. To decelerate it will use a ramp of 1000rpm/s while the maximum torque is set at the 100% of the rated value.

ControlDWord = 0x02000003
TargetTorque =1000
TargetVelocity = 6405
Acceleration = 500
Deceleration = 1000
TargetPosition = … (ignored)

Note that in the *Velocity* mode the *TargetPosition* register is ignored.

6.7.3 Torque (tq)

The T*orque* mode can be used only when the motor is equipped with encoder and allows you to control the torque available to the motor shaft. The master controller can set the torque by updating the *TargetTorque* register. The torque variation on the motor shaft always occurs according to the ramp set through the *Torque*S*lope* register.

Usually in this mode the motor speed is limited only by the maximum allowed by the motor itself, however by the *Mode_PTCNF* register it is possible to activate the speed control through the *TargetVelocity* register.

To select the *Torque* mode, simply set the bit26..24 of *ControlDWord* to the value 3 (bit26=0, bit25=1 and bit24=1).

The torque target can be processed in real or delayed time whenever the trigger is activated, as described in chapter 6.6.5 *Real-time or delayed target process*.

The conditions of achieved target reach and halted motor can be parameterized, as described in chapter 6.6.7 *Target reached and motor halted*.

Once the parameters have been set, it is possible to use the *TargetTorque* register to control the torque of the motor.

By setting the registers as shown in the table below, the torque of the motor will be set at the 50% of the rated value.

ControlDWord = 0x03000003
TargetTorque =500
TargetVelocity = … (ignored)
Acceleration = … (ignored)
Deceleration = … (ignored)
TargetPosition = … (ignored)

Note that in the *Torque* mode the *TargetVelocity*, *TargetPosition*, *Acceleration* and *Deceleration* registers are ignored.

6.7.4 Homing (hm)

Through the Homing mode the drive is able to find the zero position (also called reference). It is possible to choose among various homing methods which make use of limit switches (right and left), home switches, encoder index pulse or a combination of them.

To select the *Homing* mode, simply set the bit26..24 of *ControlDWord* to the value 7 (bit26=1, bit25=1 and bit24=1).

To perform the homing procedure, the master controller must configure the homing method through the *HomingMethod* register and then start the homing by activating the *Trigger* (transition 0->1 of bit9 of *ControlDWord).*

If the homing ends successfully, the bit10 *ReferenceSet* of Status*DWord* is set to 1 and the *ActualPosition* register is set to the value of the *HomingOffset* register. Subsequent modifications to the *HomingOffset* register do not modify the homing reference.

HomingSpeedForSwitch and *HomingSpeedForZero* registers allow you to set the speed used during the sensor search and, successively, for the homing.

The sensors connected to the drive must be associated with their corresponding digital input so that the drive can correctly read the signals and successfully complete the homing procedure. To associate the sensors with the inputs, use the registers *Home_DIA*, *NegativeLimit_DIA* and *PositiveLimit_DIA.*

The following table describes the available homing modes and the sensors used by each of them. The abbreviations used have the following meaning: **PLS**=positive limit switch, **NLS**=negative limit switch, **HS**=homing switch, **IDX**=index.

Homing methods selectable through the *HomingMethod* register					
Code	Description	Sensors used			
		PLS	NLS	HS	IDX
-80	Azzeramento sulla posizione letta dall'encoder SSI.				
1	At the start, if negative limit switch inactive counterclockwise direction up to the limit switch, then reverse and homing at the first index outside the negative limit switch. At the start, if negative limit switch active clockwise direction up to leave the limit switch, then homing at the first index outside the negative limit switch.	●			●

2	At the start, if positive limit switch inactive clockwise direction up to the limit switch, then reverse and homing at the first index outside the positive limit switch. At the start, if positive limit switch active counterclockwise direction up to leave the limit switch, then homing at the first index outside the positive limit switch.		●		●
3	At the start, if home switch inactive initial direction clockwise up to home switch, then reverse and homing at the first index outside the home switch. At the start, if home switch active initial direction counterclockwise up to leave the switch, then homing at the first index outside the home switch.			●	●
4	At the start, if home switch inactive initial direction clockwise up to home switch, then homing at the first index inside the home switch. At the start, if home switch active initial direction counterclockwise up to leave the switch, then reverse and homing at the first index inside the home switch.			●	●
5	At the start, if home switch active initial direction clockwise up to leave the switch, then homing at the first index outside the home switch. At the start, if home switch inactive initial direction counterclockwise up to find the switch, then reverse and homing at the first index outside the home switch.			●	●
6	At the start, if home switch active initial direction clockwise up to leave the switch, then reverse and homing at the first index inside the home switch. At the start, if home switch inactive initial direction counterclockwise up to find the switch, then reverse and homing at the first index inside the home switch.			●	●
7	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, reverse and homing at the first index outside the switch. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then homing at the first index outside the switch. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then homing at the first index outside the switch.	●		●	●
8	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, homing at the first index inside the home switch. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then reverse up to find again the home switch and finally homing at the first index inside the switch. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then reverse and homing at the first index inside the switch.	●		●	●
9	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, continue up to leave it then reverse and homing at the first index inside the switch. In case of positive limit switch, reverse up to find the home switch, then homing at the first index inside the switch. At the start, if home switch active initial direction clockwise up to leave	●		●	●

	the home switch, then reverse and homing at the first index inside the home switch.				
10	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, continue up to leave it then homing at the first index outside the home switch. In case of positive limit switch, reverse up to find the home switch, the reverse up to leave the switch and homing at the first index outside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then homing at the first index outside the home switch.	●		●	●
11	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, reverse and homing at the first index outside the home switch. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch and then homing at the first index outside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then homing at the first index outside the home switch.		●	●	●
12	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, homing at the first index inside the home switch. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch and then reverse up to find again the home switch and finally homing at the first index inside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the first index inside the home switch.		●	●	●
13	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then reverse and homing at the first index inside the switch. In case of negative limit switch, reverse up to find the home switch, then homing at the first index inside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the first index inside the home switch.		●	●	●
14	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then homing at the first index outside the home switch. In case of negative limit switch, reverse up to find the home switch, reverse up to leave the switch and then homing at the first index outside the switch. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then homing at the first index outside the switch.		●	●	●
17	At the start, if negative limit switch inactive initial direction counterclockwise up to find the limit switch, then reverse and homing at the active/inactive switch transition. At the start, if negative limit switch active initial direction clockwise with homing at the active/inactive switch transition.		●		
18	At the start, if positive limit switch inactive initial direction clockwise up	●			

	to the limit switch, then reverse and homing at the active/inactive switch transition. At the start, if negative limit switch active initial direction counterclockwise with homing at the active/inactive switch transition.				
19	At the start, if home switch inactive initial direction clockwise up to the home switch, then reverse and homing at the active/inactive switch transition. At the start, if home switch active initial direction counterclockwise with homing at the active/inactive switch transition.			●	
20	At the start, if home switch inactive initial direction clockwise up to the home switch, then homing at the inactive/active switch transition. At the start, if home switch active initial direction counterclockwise up to leave the switch, then reverse and homing at the inactive/active switch transition.			●	
21	At the start, if home switch active initial direction clockwise up to leave the switch, then homing at the active/inactive switch transition. At the start, if home switch inactive initial direction counterclockwise up to find the switch, then reverse and homing at the active/inactive switch transition.			●	
22	At the start, if home switch active initial direction clockwise up to leave the switch, then reverse and homing at the inactive/active switch transition. At the start, if home switch inactive initial direction counterclockwise up to find the switch, then homing at the inactive/active switch transition.			●	
23	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, reverse and homing at the active/inactive switch transition. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then homing at the active/inactive switch transition. At the start, if home switch active initial direction counterclockwise up to leave the switch, then homing at the active/inactive switch transition.	●		●	
24	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, homing at the inactive/active switch transition. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then reverse up to find again the home switch and finally homing at the inactive/active switch transition. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.	●		●	
25	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, continue up to leave it, then reverse and homing at the inactive/active switch transition. In case of positive limit switch, reverse up to find the home switch, homing at the inactive/active switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.	●		●	
26	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, continue up to leave it, then homing at the active/inactive switch	●		●	

	transition. In case of positive limit switch, reverse up to find the home switch, then reverse up to leave the switch and finally homing at the active/inactive switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then homing at the active/inactive switch transition.				
27	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, reverse and homing at the active/inactive switch transition. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch, then homing at the active/inactive switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then homing at the active/inactive switch transition.		●	●	
28	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, homing at the inactive/active switch transition. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch, then reverse up to find again the home switch and finally homing at the inactive/active switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.		●	●	
29	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then reverse and homing at the inactive/active switch transition. In case of negative limit switch, reverse up to find the home switch, then homing at the inactive/active switch transition. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.		●	●	
30	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then homing at the active/inactive switch transition. In case of negative limit switch, reverse up to find the home switch, then reverse up to leave the switch, then homing at the active/inactive switch transition. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then homing at the active/inactive switch transition.		●	●	
33	Initial direction counterclockwise with homing at the first index found.				●
34	Initial direction clockwise with homing at the first index found.				●
37	Homing at the actual position.				

6.8 Mapping of Registers

Many Modbus functions can access more contiguous registers at the same time (just think, for example, of the *Write Multiple Registers* function), but if the registers do not have a sequential address it is necessary to repeat the write or read command several times.

The mapping function allows to assign a second address to the registers in order to create a new sequence of registers, one after the other, to be able to access them with a single read and write operation, with obvious benefits in terms of communication speed and mains traffic reduction.

The address area in which the registers can be mapped is between 40801 and 40896 and allows to map up to 96 registers.

To map a register to a specific position, simply write the address in the corresponding *MappingRegister_nn_RMP*. The *MappingRegister_1_RMP* is associated with the address 40801, the *MappingRegister_2_RMP* is associated with the address 40802 and so on, therefore, for example, if you want to map a particular register to the address 40822 simply write the address in the *MappingRegister_22_RMP*.

If the register you want to map has a dimension of 4 bytes (two words, 32bit) it is necessary to use two consecutive mapping registers both filled with the address of the register to be mapped. For example, by writing in both mapping registers MappingRegister_8_RMP and MappingRegister_9_RMP the value 42914, corresponding to the *TargetVelocity* register of dimension 32bit, it will be possible to read or write *TargetVelocity* also at the address 40808, as well as through the native address 42914.

7 Errors and diagnostics

The drive is equipped with 3 LEDs, positioned in the front, which provide information on the status of the power supply, the motor enabling and the presence of errors.

7.1 LED *On*

The green On LED provides information on the status of the power supply, as indicated in the table below:

Led *On*	
Off	Driver not supplied
On	Driver supplied

7.2 LED *En*

The yellow En LED provides information on the status of the motor, as indicated in the table below:

Led *En*	
Off	Motor not enabled
Fixed on	Motor enabled stopped
Flashing	Motor enabled in rotation

7.3 LED *Fault*

The red Fault LED is used to indicate presence of errors in the drive, as shown in the table below:

Led *Fault*	
Off	No error
Fixed on	Corrupt firmware. Active Loader.
Flashing	Presence of errors. The number of flashes indicates the type of error as detailed below.

7.4 Errors

The drive is able to detect many error conditions and to intervene by stopping or disabling the motor.

The error condition is signaled by the red LED on the front panel of the drive, through the digital outputs and the fieldbus.

The errors are divided into classes. Each class identifies a specific reaction of the drive, according to the table below:

Class	Description	Drive's reaction
0	Warning, there is no impediment to continue with the operations in progress.	None.
2	Error which requires the stop of the motor but not the transition to FAULT.	Deceleration with Quick Stop.
4	Error which requires the stop of the motor and successively the transition to FAULT.	Deceleration with Quick Stop and subsequent motor disabling.
6	Error which requires the immediate disabling of the motor and the transition to FAULT.	Motor disabling.
8	As per class 6. The error can be reset only through a cycle of turning off and on.	Motor disabling.

For error class 8, note that the power off/on cycle must consider the eventual auxiliary power supply and the connection to the DUP port, which can keep the logic part supplied and thus prevent the reset of the drive.

Each error has a code which identifies the type and in some case a sub-code which helps to identify the origin of the problem. The error code is reported in the *ErrorCode* register while the sub-code in the *ErrorSubCode* register.

The errors are grouped by affinity and displayed to the user through a different number of flashes of the red LED.

The following table summarizes the errors recognized by the driver:

<table>
<tr><th>Error code</th><th>Class</th><th>LED Flash</th><th>bit</th><th>Description / Sub-code</th></tr>
<tr><td>2310$_h$</td><td>6</td><td>4</td><td>3</td><td>Detected motor over current
<table>
<tr><th>Sub-code</th><th>Description</th></tr>
<tr><td>0100$_h$</td><td>During PWM modulation</td></tr>
<tr><td>0101$_h$</td><td>Towards ground</td></tr>
<tr><td>0102$_h$</td><td>Towards positive power bus</td></tr>
<tr><td>0103$_h$</td><td>Short circuit between phase A+ and ground</td></tr>
<tr><td>0104$_h$</td><td>Short circuit between phase A- and ground</td></tr>
<tr><td>0105$_h$</td><td>Short circuit between phase B+ and ground</td></tr>
<tr><td>0106$_h$</td><td>Short circuit between Phase B- and ground</td></tr>
<tr><td>0109$_h$</td><td>Abnormal current between phase A+ and ground</td></tr>
<tr><td>010A$_h$</td><td>Abnormal current between phase A- and ground</td></tr>
<tr><td>010B$_h$</td><td>Abnormal current between phase B+ and ground</td></tr>
<tr><td>010C$_h$</td><td>Abnormal current between phase B- and</td></tr>
</table>
</td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td></td><td><table><tr><td></td><td>ground</td></tr></table></td></tr>
<tr><td>23A0$_h$</td><td>6</td><td>5</td><td>5</td><td>Detected motor phase open (not connected or interrupted)<table><tr><td>Sub-code</td><td>Description</td></tr><tr><td>0201$_h$</td><td>Phase A</td></tr><tr><td>0202$_h$</td><td>Phase B</td></tr></table></td></tr>
<tr><td>23B2$_h$</td><td>6</td><td>8</td><td>4</td><td>Detected over current on the encoder power supply +V</td></tr>
<tr><td>3210$_h$</td><td>6</td><td>2</td><td>1</td><td>Power supply voltage higher than the maximum allowed</td></tr>
<tr><td>3220$_h$</td><td>6</td><td>1</td><td>0</td><td>Power supply voltage lower than the minimum necessary for a correct operation</td></tr>
<tr><td>4210$_h$</td><td>4</td><td>3</td><td>2</td><td>Power stage temperature higher than the maximum allowed value</td></tr>
<tr><td>5592$_h$</td><td>8</td><td>13</td><td>24</td><td>Invalid device descriptor</td></tr>
<tr><td>5594$_h$</td><td>8</td><td>13</td><td>24</td><td>Invalid device configuration</td></tr>
<tr><td>55A0$_h$</td><td>8</td><td>13</td><td>31</td><td>Inappropriate installed firmware</td></tr>
<tr><td>55A2$_h$</td><td>8</td><td>13</td><td>31</td><td>Incompatible installed firmware revision</td></tr>
<tr><td>6200$_h$</td><td>8</td><td>15</td><td>31</td><td>Firmware error</td></tr>
<tr><td>63A0$_h$</td><td>6</td><td>10</td><td>16</td><td>Conflicts in the configuration<table><tr><td>Sub-code</td><td>Description</td></tr><tr><td>0701$_h$</td><td>Selected Torque mode without encoder feedback (present or configured).</td></tr><tr><td>0702$_h$</td><td>Selected current proportional to the load (Mode_CRRG register) without encoder feedback (present o configured).</td></tr><tr><td>0704$_h$</td><td>Activated encoder feedback without having activated or configured the encoder</td></tr><tr><td>0705$_h$</td><td>Activated the following error control without having activated or configured the encoder selected as feedback source.</td></tr><tr><td>0706$_h$</td><td>Activated the incremental encoder Index management and contemporaneously the SSI encoder.</td></tr></table></td></tr>
<tr><td>7282$_h$</td><td>8</td><td>14</td><td>11</td><td>Detected invalid offset values</td></tr>
<tr><td>8100$_h$</td><td>6</td><td>7</td><td>28</td><td>Bus communication error</td></tr>
<tr><td>82D6$_h$</td><td>6</td><td>7</td><td>22</td><td>The client device did not reset the client watchdog in the expected time. The indicated class error expresses the default value. The class can be modified through the ClientWatchDogOption_WDOG register.</td></tr>
<tr><td>82D7$_h$</td><td>6</td><td>7</td><td>22</td><td>It occurs when the characters received are not contiguous but interrupted with a delay that violates the Modbus RTU specification. For more details see chapter 2.5.1.1 MODBUS Message RTU Framing of the document “MODBUS over Serial Line Specification and Implementation Guide V1.02” available on the official website www.modbus.org. The error class can be modified through the ComunicationOption_FBCNF register.</td></tr>
<tr><td>8611$_h$</td><td>6</td><td>6</td><td>18</td><td>Following error. The indicated error class expresses the default value. The class can be modified through the Following_Error_ERRCS error.</td></tr>
</table>

<table>
<tr><td>8612_h</td><td>2</td><td>6</td><td>19</td><td>Limit switch activation
<table><tr><td>Sub-code</td><td>Description</td></tr><tr><td>0801_h</td><td>Positive limit activation</td></tr><tr><td>0802_h</td><td>Negative limit activation</td></tr><tr><td>0803_h</td><td>Positive movement with positive limit switch active</td></tr><tr><td>0804_h</td><td>Negative movement with negative limit switch active</td></tr></table></td></tr>
<tr><td>8613_h</td><td>2</td><td>6</td><td>17</td><td>Homing error
<table><tr><td>Sub-code</td><td>Description</td></tr><tr><td>0601_h</td><td>Index not found</td></tr><tr><td>0602_h</td><td>Unexpected clockwise limit switch</td></tr><tr><td>0603_h</td><td>Unexpected counterclockwise limit switch</td></tr><tr><td>0604_h</td><td>The selected homing mode uses the incremental encoder Index but the management of the Index itself is not active (Configuration_ENCMTR register)</td></tr><tr><td>0605_h</td><td>The selected homing mode uses the SSI encoder but the SSI encoder reading is not operative.</td></tr></table></td></tr>
</table>

8 Registers

The device is parameterized and controlled through the writing and reading of registers. Some registers are read-only accessible while others can be both read and written. Some of them can be modified only in certain status of the device. For example the *Resistance_MTRDT* register which allows to set the motor phase resistance, cannot be modified if the drive is enabled.

Registers are accessed by one of the functions described in chapter 4.2 Supported Services and Features.

8.1 Device Identification

Address	Name	Mnemonic
40101	*Product code*	

Data Type	Access Type	PDO Mapping	Note
u32	ro		

Default Value	Minimum	Maximum	Unit

Description

Contains the unique device specific code.

The following table shows the correspondence between code and device:

Codice	Dispositivo	Codice	Dispositivo
00504100$_h$	DDS5041	00504101$_h$	DDS5041A
00504400$_h$	DDS5044	00504401$_h$	DDS5044A
00504800$_h$	DDS5048	00504801$_h$	DDS5048A
00507400$_h$	DDS5074	00507401$_h$	DDS5074A
00507800$_h$	DDS5078	00507801$_h$	DDS5078A
00524100$_h$	DDS5241	00524101$_h$	DDS5241A
00524400$_h$	DDS5244	00524401$_h$	DDS5244A
00524800$_h$	DDS5248	00524801$_h$	DDS5248A
00527400$_h$	DDS5274	00527401$_h$	DDS5274A
00527800$_h$	DDS5278	00527801$_h$	DDS5278A
00544100$_h$	DDS5441	00544101$_h$	DDS5441A
00544400$_h$	DDS5444	00544401$_h$	DDS5444A
00544800$_h$	DDS5448	00544801$_h$	DDS5448A
00547400$_h$	DDS5474	00547401$_h$	DDS5474A
00547800$_h$	DDS5478	00547801$_h$	DDS5478A

Address	Name	Mnemonic
40103	*Serial number*	

Data Type	Access Type	PDO Mapping	Note
u32	ro		

Default Value	Minimum	Maximum	Unit

Description

Contains the device serial number.

Address	Name			Mnemonic
40105	*Hardware revision*			
	Data Type	Access Type	PDO Mapping	Note
	u32	ro		
	Default Value	Minimum	Maximum	Unit

Description

Contains the device Hardware revision.

The high word contains the revision of the special versions and the low word contains the standard product revision. Each revision is organized into Major revision and Minor revision.

Special Hardware Revision		Hardware Revision	
Bits 31..24	Bits 23..16	Bits 15..8	Bits 7..0
Major revision number	Minor revision number	Major revision number	Minor revision number

Address	Name			Mnemonic
40107	*Firmware revision*			
	Data Type	Access Type	PDO Mapping	Note
	u32	ro		
	Default Value	Minimum	Maximum	Unit

Description

Contains the device Firmware revision.

The high word contains the revision of the special versions and the low word contains the standard product revision. Each revision is organized into Major revision and Minor revision.

Special Hardware Revision		Hardware Revision	
Bits 31..24	Bits 23..16	Bits 15..8	Bits 7..0
Major revision number	Minor revision number	Major revision number	Minor revision number

8.2 Store Parameters

It allows to save many dictionary registers in the non-volatile memory of the device. The dictionary registers that can be saved are marked by the symbol 🖫 in the field *Note*.

When the value of a register is saved in the non-volatile memory, it is automatically restored at the power on or in case of device reset.

At most it is possible to save or restore the default values for 10.000 times. The saving is completed in about 100ms.

Address	Name			Mnemonic
40111	*SaveAllParameters_SPF*			
	Data Type	Access Type	PDO Mapping	Note
	u32	rw		The default values can be saved or restored only with the motor disabled. Trying the operation with the motor enabled, an error code is obtained in response.
	Default Value	Minimum	Maximum	Unit
	Description			
	It allows to save the dictionary registers, marked by the symbol 🖫, in the non-volatile memory. To start and save simply write the register with the value **65766173**$_h$.			

8.3 Restore Default Parameters

It allows to restore the default value for the registers saved using the *Store Parameters* function.

At most it is possible to restore the default values for 10.000 times. The restoring is completed in about 100ms.

Address	Name			Mnemonic
40121	*RestoreAllDefaultParameters_RDP*			
	Data Type	Access Type	PDO Mapping	Note
	u32	rw		The default values can be restored only with the motor disabled. Trying the operation with the motor enabled, an error code is obtained in response.
	Default Value	Minimum	Maximum	Unit
	Description			
	It allows to restore the default value for the dictionary registers. To start to restore simply write the register with the value **64616F6C**$_h$.			

8.4 Modbus RTU communication setting

It is possible to activate the characteristics of the Modbus RTU communication to the application by acting on the registers described below. It is also possible to activate supervision and control mechanisms so that the interruption of communication is detected by the drive as an error condition.

Address	Name			Mnemonic
40206	*ComunicationOption_FBCNF*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		
	Default Value	**Minimum**	**Maximum**	**Unit**
	41_h			

Description

It allows to define the reaction of the device in case of communication problems.

The lower 4 bits of the low byte of the word encode the reaction of the device when the motor is not enabled, while the higher 4 bits of the low byte of the word define the reaction when the motor is enabled.

Bit8 CloseSocket if set to 1 causes the port to close in case of Ethernet link failure.

<table>
<tr><th colspan="4">word</th></tr>
<tr><th colspan="2">High byte</th><th colspan="2">Low byte</th></tr>
<tr><th>bits 15..12</th><th>bits 11..8</th><th>bits 7..4</th><th>bits 3..0</th></tr>
<tr><td>Not used, set to 0</td><td><table><tr><th>bit</th><th></th></tr><tr><td>11</td><td rowspan="3">Non used, set to 0</td></tr><tr><td>10</td></tr><tr><td>9</td></tr><tr><td>8</td><td>CloseSocket</td></tr></table></td><td>Reaction with motor enabled</td><td>Reaction with motor disabled</td></tr>
</table>

The following table shows the possible values for the bits 7..4 and bit 3..0:

bit 7..4 bit 3..0	Device reaction
0	No reaction.
1	Display only.
2	Deceleration with Quick Stop.
3	Deceleration with Quick Stop and subsequent motor disabling.
4	Motor disabling.

For example, by setting the register with the value 41_h, in the event of a communication error there will be a simple signal with the motor disabled, while with the motor enabled it will cause the disabling of the motor and an error message.

8.5 Client Watch Dog

<table>
<tr><td rowspan="7">Address
40241</td><td colspan="3">Name
ClientWatchDogOption_WDOG</td><td>Mnemonic</td></tr>
<tr><td>Data Type
u16</td><td>Access Type
rw</td><td>PD Mapping</td><td>Note
It cannot be modified if bit *Lock* set to 1.</td></tr>
<tr><td>Default Value
40_h</td><td>Minimum</td><td>Maximum</td><td>Unit</td></tr>
<tr><td colspan="4">Description</td></tr>
</table>

It allows to define the reaction of the device in case of time-out of the client watchdog.

The lower 4 bits of the low byte of the word encode the reaction of the device when the motor is not enabled, while the higher 4 bits of the low byte of the word define the reaction when the motor is enabled.

Bit8 CloseSocket if set to 1 causes the port to close in case of Ethernet link failure.

The bit9 Lock is useful to prevent accidental modifications to the client watchdog and, if set to 1, it blocks any other modification of the *ClientWatchDogOption_WDOG* and *ClientWatchDogTimeOut_WDOG* register. The bit can be reset to 0 only through the device reset or by a power off/on cycle.

<table>
<tr><th colspan="4">word</th></tr>
<tr><th colspan="2">High byte</th><th colspan="2">High byte</th></tr>
<tr><th>bits 15..12</th><th>bits 11..8</th><th>bits 7..4</th><th>bits 3..0</th></tr>
<tr><td>Not used, set to 0</td><td><table><tr><th>bit</th><th></th></tr><tr><td>11</td><td>Not used, set to</td></tr><tr><td>10</td><td>0</td></tr><tr><td>9</td><td>Lock</td></tr><tr><td>8</td><td>CloseSocket</td></tr></table></td><td>Reaction with motor enabled</td><td>Reaction with motor disabled</td></tr>
</table>

The following table shows the possible values for the bits 7..4 and bit 3..0:

bit 7..4 bit 3..0	Device reaction
0	No reaction.
1	Display only.
2	Deceleration with Quick Stop.
3	Deceleration with Quick Stop and subsequent motor disabling.
4	Motor disabling.

	For example, by setting the register with the value 40_h, the watchdog time-out will have no effect with the motor disabled, while with the motor enabled it will cause the disabling of the motor and an error message.

Address	Name			Mnemonic
40242	*ClientWatchDogTimeOut_WDOG*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		🖫 It cannot be modified if bit *Lock* set to 1. With the motor enabled, it can be modified only if the previous value was 0.
	Default Value	Minimum	Maximum	Unit
	0			1ms (Ex. 10 = 10ms)
	Description			
	It allows to set the maximum time within which the client must reset the watchdog by writing the *WatchDogReload* register. If the *WatchDogReload* register is not written by the client within the time set by *ClientWatchDogTimeOut_WDOG,* a time-out occurs and the device reacts as set through th *ClientWatchDogOption_WDOG* register. By setting the *ClientWatchDogTimeOut_WDOG* register to 0, the client watchdog is disabled.			

Address	Name			Mnemonic
40295	*WatchDogReload*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		
	Default Value	Minimum	Maximum	Unit
	Description			
	When this register is written with the value **A3C5**$_h$, the client watchdog is reset. In reading the registers always returns the value $5C3A_h$.			

8.6 Mapping of Registers

Thanks to the mapping of registers, it is possible to arrange the most used registers in sequence so as to be able to access them sequentially with a single read or write command.

A mapped register still maintains the native address but a second address is added, within the mapping area, through which it can be read or written.

The address area in which the registers can be mapped is between 40801 and 40896 while the mapping order is defined by the contents of the *MappingRegister_nn_RMP* registers described below.

<table>
<tr><td rowspan="4">Address
40601…
40696
(96 registers)</td><td colspan="3">Name
MappingRegister_1_RMP … MappingRegister_96_RMP</td><td>Mnemonic</td></tr>
<tr><td>Data Type
u16</td><td>Access Type
rw</td><td>PD Mapping</td><td>Note</td></tr>
<tr><td>Default Value
0</td><td>Minimum</td><td>Maximum</td><td>Unit</td></tr>
<tr><td colspan="4">Description

In total, 96 *MappingRegister_nn_RMP* registers are available for mapping just as many registers in the area between 40801 and 40896.

The mapping registers *MappingRegister_nn_RMP* have base address 40601 while the mapped registers have base address 40801. The correspondence between the mapping register and the mapped register is one-to-one, therefore setting for example the mapping register *MappingRegister_1_RMP*, with address 40601, with the value 42913 (that corresponds to the address of the *TargetTorque* register with length 16bit), it will be possible to access the mapped register even through the address 40801, as well as through the native address 42913.

If the register you want to map has a dimension of 4 bytes (two words, 32bit) it is necessary to use two consecutive mapping registers both filled with the address of the register to be mapped. For example, by writing in both mapping registers MappingRegister_8_RMP and MappingRegister_9_RMP (respectively with address 40608 and 40609) the value 42914, corresponding to the *TargetVelocity* register of dimension 32bit, it will be possible to read or write *TargetVelocity* also through the address 40808, as well as through the native address 42914.

By writing the value 0 in the mapping registers, the corresponding 40801 base address becomes inaccessible.</td></tr>
</table>

8.7 Faults and Errors setting

Address	Name			Mnemonic
41221	*OpenPhase_EFSTP*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	u8	rw		
	Default Value	**Minimum**	**Maximum**	**Unit**
	1			

Description

It allows to enable or disable the open phase condition test.

The following table shows the possible values:

0	Open phase test disabled
1	Open phase test enabled

8.8 Motor Data

Address	Name			Mnemonic
41311	*CMC_MTRDT*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u32	rw		It cannot be modified with drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			

Description

It is a compact code that identifies the LAM Technologies motors and that allows the drive to configure the best control parameters for the motor.

It is sufficient to initialize the *CMC Motor Data* register with the CMC code of the motor connected to the drive to instruct the drive itself about all motor features.

When the register is initialized with a value different from 0, the registers from 4312_h to $431C_h$ are ignored.

Address	Name			Mnemonic
41313	*Type_MTRDT*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u8	ro		Valid only if *CMC_MTRDT* = 0
	Default Value	**Minimum**	**Maximum**	**Unit**
	12			

Description

It indicates the type of motor connected to the drive according to the table below:

12	Two-phase Stepper Motor

Address	Name			Mnemonic
41314	*PolePairs_MTRDT*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u8	rw		Valid only if *CMC_MTRDT* = 0. It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	50	50	50	
	Description			
	It allows to set the poles number of the motor. The number of poles is the number of electrical cycles included in a complete motor revolution. For example, a 200 steps/revolution two-phase stepper motor has a number of poles equal to 50.			

Address	Name			Mnemonic
41315	*Wiring_MTRDT*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u8	ro		Valid only if *CMC_MTRDT* = 0
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			
	Description			
	Motor connection. Currently not used.			

Address	Name			Mnemonic
41316	*Resistance_MTRDT*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		Valid only if *CMC_MTRDT* = 0. It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	100		60000	10mOhm (Ex. 240 = 2.4 Ohm)
	Description			
	It allows to set the phase resistance of the motor connected to the drive. The set value must take into account the type of phases connection (series or parallel), if the motor allows multiple configuration.			

Address	Name			Mnemonic
41317	*Inductance_MTRDT*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		Valid only if *CMC_MTRDT* = 0It cannot be modified with drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	300		60000	10µH (Ex. 320 = 3.2 mH)
	Description			
	It allows to set the phase inductance of the motor connected to the drive. The set value must take into account the type of phase connection (series or parallel), if the motor allows multiple configurations. For an optimal operation it is very important to carefully set this register to reflect the actual inductance of the connected motor.			

Address	Name			Mnemonic
41318	*BackEMF_MTRDT*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		Valid only if *CMC_MTRDT* = 0. It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	2500		60000	10mV/1000rpm (Ex. 4500 = 45V/1000rpm)
	Description			
	It allows to set the BEMF of the motor connected to the drive. The set value must take into account the type of phase connection (series or parallel), if the motor allows multiple configurations.			

Address	Name			Mnemonic
41319	*RatedCurrent_MTRDT*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		Valid only if *CMC_MTRDT* = 0. It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	100	10	60000	10mArms (Ex. 420 = 4.2 Arms)
	Description			
	It allows to set the rated current of the motor connected to the drive. The set value must take into account the type of phase connection (series or parallel), if the motor allows multiple configurations. For an optimal operation it is very important that the setting of this register is accurate and reflects the actual value of the rated current with which the motor was built. Do not use this register to modify the motor current. The *RatedCurrent_MTRDT* register must always be set with the rated current value indicated by the motor manufacturer. To modify the running or idle current of the motor, please use instead the *CurrentMax_MTRCNF* and *CurrentMin_MTRCNF* registers. If the set current value exceeds the maximum device capacity, the latter shall prevail.			

Address	Name			Mnemonic
41320	*MaxCurrent_MTRDT*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		Valid only if *CMC_MTRDT* = 0. It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	130	10	60000	10mArms (Ex. 550 = 5.5 Arms)
	Description			
	It allows to set the maximum current applied to the motor for short periods.			

	The set value must take into account the type of phase connection (series or parallel), if the motor allows multiple configurations. In case of a LAM Technologies motor, the *MaxCurrent_MTRDT* register can be set at a value equal to the 130% of the parameter *RatedCurrent_MTRDT.* For different motors it is necessary to ask the maximum permissible current to the motor's manufacturer. If the set current value exceeds the maximum device capacity, the latter shall prevail.

Address	Name			Mnemonic
41321	*RatedTorque_MTRDT*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		🖫 Valid only if *CMC_MTRDT* = 0. It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	50	1	60000	10mNm (Ex. 180 = 1,8 Nm)
	Description			
	It allows to set the rated torque of the motor connected to the drive. For optimal operation it is very important that the setting of this register is accurate and reflects the motor rated torque when supplied at the rated current set through the *RatedCurrent_MTRDT* register.			

Address	Name			Mnemonic
41322	*MaxSpeed_MTRDT*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		🖫
	Default Value	Minimum	Maximum	Unit
	30000	1	30000	0.1rpm (Ex. 668 = 66.8rpm)
	Description			
	The register sets the maximum motor speed. This value can never be exceeded and prevails over any other setting, whatever the chosen operation mode (also *Torque mode*).			

8.9 Motor Incremental Encoder Data

Address	Name			Mnemonic
41331	*Configuration_ENCMTR*			
	Data Type	Access Type	PD Mapping	Note
	u8	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	0			

Description

It allows you to set the encoder management mode. The register is made up of groups of bits that define the following functions:

Bits	Functions
0	Index 0 – Index signal not managed 1 – Index signal managed
1	Pulse count inversion 0 – Normal pulse count 1 – Backwards pulses count (equivalent to change phase A and B between them)
2	Not used
3	Power Supply 0 – Encoder power supply off 1 – Encoder power supply on
7..4	Filter If the encoder signals are affected by electric noise, it is possible to use the digital filter present in the drive to eliminate the pulses shorter than a pre-defined value. 3 – 250ns 5 – 500ns 7 – 1us 9 – 2us 12 – 4us 15 – 8us 0, 1, 2, 4, 6, 8, 10, 11, 13, 14 – Reserved values, do not use The maximum equivalent frequency must be calculated considering twice the set time, as this time must elapse both for the active signal condition and for the inactive signal. For example, setting a Filter value of 1us you can apply a maximum frequency of 500KHz (1+1=2us i.e. 500KHz).

	If the register is set to 0, the encoder input is disabled.

Address	Name			Mnemonic
41332	*CPR_ENCMTR*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			Encoder pulses per revolution.
	Description			
	It allows to set the pulses number/revolution of the encoder connected to the motor. By setting the value to 0, the encoder input is disabled. The *CPR_ENCMTR* register must be configured with the value 0 if the encoder is not present or used. The drive is able to count each pulse edge in order to obtain a resolution 4 times higher than the native resolution of the encoder. For example, by using a 400 pulses/revolution encoder, the drive will be able to recognize 1600 different positions/rev. The encoder must be able to generate at least 360 pulses /rev.			

The described *Configuration_ENCMTR* and *CPR_ENCMTR* registers are automatically updated each time the *CMC_MTRDT* register is written with a valid LAM Technologies motor code.

Address	Name			Mnemonic
41333	*PhasingTime_ENCMTR*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		🖫
	Default Value	**Minimum**	**Maximum**	**Unit**
	500			1 ms (1500 = 1.5ms)
	Description			
	It allows you to set the waiting time to phase the encoder. The first time that a motor equipped with an encoder is enabled, the encoder must be phased. When enabled, the motor could move and it is necessary to wait for the load to stabilize before proceeding to phase the encoder. The *PhasingTime_ENCMTR* register allows you to set this time.			

8.10 SSI Encoder Data

Address	Name			Mnemonic
41351	*Configuration_ENCSSI*			
	Data Type	Access Type	PDO Mapping	Note
	u8	rw		It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	0			

Description

It allows you to enable the SSI encoder and sets its management mode. The register is made up of groups of bits that define the following functions:

Bits	Functions
0	Enabling 0 - SSI Encoder reading not enabled 1 - SSI Encoder reading enabled
1	Error checking 0 - Error control not enabled 1 - Error control enabled When the error control is enabled, the drive signals an error condition if the SSI encoder reading returns all the bits to 1, all the bits to 0 or if the sampled bit on the first falling edge of the clock differs from 1.
2	Position copy 0 - Copy not enabled 1 - Copy enabled When the position copy is enabled, the SSI Encoder quota is copied in the actual motor position (*Position_actual_value* register) the first time the quota becomes valid and at each transition to the *Operational* status of the drive. The actual position of the motor can be initialized to the quota read by the SSI encoder also through the *Homing* operation mode by selecting the mode -80.
3	Power supply 0 - SSI encoder power supply off 1 - SSI encoder power supply on
7..4	Not used. Set to 0.

If the register is set to 0, the SSI encoder input is disabled.

Address	Name			Mnemonic
41352	*FrameLength_ENCSSI*			
	Data Type	Access Type	PDO Mapping	Note
	u8	rw		It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	25	8	47	bit
	Description			

It allows to set the number of bits that make up the data frame of the SSI encoder. The *start* bit must not be counted.

Address	Name			Mnemonic
41353	*TurnsBits_ENCSSI*			
	Data Type	Access Type	PDO Mapping	Note
	u8	rw		It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	12		24	bit
	Description			

It allows to set the number of bits reserved the full revolutions counting.

In case of linear SSI encoder, set the register to 0.

Address	Name			Mnemonic
41354	*SingleTurnBits_ENCSSI*			
	Data Type	Access Type	PDO Mapping	Note
	u8	rw		It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	13	8	31	bit
	Description			

It allows to set the number of bits reserved for the position on the revolution.
In case of a linear encoder, set the register with the number of bits that make up the quota of the encoder itself.

Address	Name			Mnemonic
41355	*MSB_Offset_ENCSSI*			
	Data Type	Access Type	PDO Mapping	Note
	u8	rw		It cannot be modified with the drive enabled.
	Default Value	Minimum	Maximum	Unit
	0		40	bit
	Description			

	It allows to set the position of the most significant bit (MSB) inside the frame. The position is counted from left to right with the fist bit of the frame that has position 0. For example, if the most significant bit of the quota falls on the third bit of the frame (starting from left and not counting the *start* bit), the *MSB_Offset_ENCSSI* register must be set to the value 3.

Address	Name			Mnemonic
41356	*FormingTime_ENCSSI*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	u16	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	150			1 us (150 = 150us)
	Description			
	It allows to set the time required by the SSI encoder to prepare a new quota. The drive waits at least this time between one reading and the next.			

Address	Name			Mnemonic
41357	*SetupTime_ENCSSI*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	u16	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	1200			1 ms (1200 = 1.2ms)
	Description			
	It allows to set the time required for the SSI encoder to complete the initializations and provide the first valid quota. After the SSI encoder is enabled or after power has been supplied (if the SSI encoder is configured with bit3 of the *Configuration_ENCSSI* register to 1), the drive waits at least this time before executing the first reading of the quota.			

Address	Name			Mnemonic
41358	*ScalingFactor_N_ENCSSI*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	i32	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	10000			
	Description			
	It allows to set the numerator of the formula used to calculate the position expressed in 0,0001rev, starting from the quota read by the SSI encoder. The formula used is the following:			

	*Position_ENCSSIV = (PositionRaw_ENCSSIV - PositionOffset_ENCSSI) * ScalingFactor_N_ENCSSI / ScalingFactor_D_ENCSSI*

Address	Name			Mnemonic
41360	*ScalingFactor_D_ENCSSI*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	i32	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	10000			
	Description			
	It allows to set the denominator of the formula used to calculate the position expressed in 0,0001rev, starting from the quota read by the SSI encoder. The formula used is the following: *Position_ENCSSIV = (PositionRaw_ENCSSIV - PositionOffset_ENCSSI) * ScalingFactor_N_ENCSSI / ScalingFactor_D_ENCSSI*			

Address	Name			Mnemonic
41362	*PositionOffset_ENCSSI*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	i32	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	10000			
	Description			
	It allows to set the offset subtracted from the quota read by the SSI encoder to calculate the position expressed in 0,0001rev. The formula applied is the following: *Position_ENCSSIV = (PositionRaw_ENCSSIV - PositionOffset_ENCSSI) * ScalingFactor_N_ENCSSI / ScalingFactor_D_ENCSSI*			

Address	Name			Mnemonic
41364	*PhasingTime_ENCSSI*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		🖫 It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	500			1 ms (1500 = 1.5ms)
	Description			
	It allows to set the time required to phase the encoder. The first time a motor equipped with encoder is enabled, the encoder must be phased.			

Following the enabling, the motor could move and it is necessary to wait for the load to stabilize before proceeding with the encoder phasing. The *PhasingTime_ENCSSI* register allows to set this time.

8.11 Holding Brake Setup

Address	Name			Mnemonic
41371	*Option_HBRKS*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	u16	rw		
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			

Description

It allows to configure the holding brake management by the drive and by manual control.
The register is made up by groups of bits that operates as follows:

<table>
<tr><th>Bits</th><th>Function</th></tr>
<tr><td>0</td><td>Handling
<table><tr><td>0</td><td>Not handled, the holding brake is not handled through the drive</td></tr><tr><td>1</td><td>Handles, the holding brake is handled through the drive (connected to one output)</td></tr></table>
When the bit is set to 1 upon enabling, the drive waits for the brake disengagement time specified by the Release_Time_HBRKS register before going into the operating status. When disabled, the drive waits for the Application_Time_HBRKS time before disconnecting the motor.</td></tr>
<tr><td>1</td><td>Automatic / Manual
<table><tr><td>0</td><td>Automatic, the drive command the activation and deactivation of the brake autonomously</td></tr><tr><td>1</td><td>Manual, the brake control can take place only via a digital input or the Control_HBRKC register</td></tr></table></td></tr>
<tr><td>2..7</td><td>Not used</td></tr>
<tr><td>8..11</td><td>Action

It allows to define how the manual brake control (obtainable through a digital input or the Control_HBRKC register) interacts with the automatic control operated by the drive (if active through bit1=0).

In the following description, “active signal” means the active state of the digital input chosen for the manual brake control or the bit0 of the Control_HBRKC register set to 1 (the two signals are in logic OR with each other).</td></tr>
</table>

0	None
1	Release, with active signal the brake is always released independently from the drive control
2	Engage, with active signal the brake is always engaged independently from the drive control
3	Release / Engage, with active signal the brake is always released. With inactive signal the brake is always engaged independently from the drive control
4	Shared Release, on the signal rising edge (transition from inactive to active) the brake is released. Also the drive can release the brake when necessary
5	Shared Engage, on the signal rising edge (transition from inactive to active) the brake is engaged. Also the drive can engage the brake when necessary
6	Shared release / engage, on the signal rising edge (transition from inactive to active) the brake is released while on the signal falling edge (transition from active to inactive) the brake is engaged. Also the drive can release or engage the brake when necessary

Address	Name			Mnemonic
41372	*Application_Time_HBRKS*			
	Data Type	Access Type	PDO Mapping	Note
	u16	rw		🖫
	Default Value	Minimum	Maximum	Unit
	200		10000	ms (ex. 250 = 250ms)

Description

It allows you to set the time required for the brake to completely engage to ensure the maximum braking torque.

When the bit0 of the *Option_HBRKS* register is set to 1 and the disabling of the motor is required, the drive waits for the time specified in the *Application_Time_HBRKS* register, after the brake has been engaged, before disconnecting the motor.

Address	Name			Mnemonic
41373	*Release_Time_HBRKS*			
	Data Type	Access Type	PDO Mapping	Note
	u16	rw		🖫
	Default Value	Minimum	Maximum	Unit
	200		10000	ms (ex. 250 = 250ms)

Description

It allows you to set the time necessary for the brake to completely disengage to ensure the minimum braking torque.

When the bit0 of the *Option_HBRKS* register is set to 1 and the motor enable is required, the drive waits for the time specified in the *Release_Time_HBRKS* register, after the brake has been disengaged, before going into the operating status.

Address	Name		Mnemonic
41374			
Data Type	Access Type	PDO Mapping	Note
u32	rw		
Default Value	Minimum	Maximum	Unit
Description			
Reserved, do not use.			

Address	Name		Mnemonic
41376			
Data Type	Access Type	PDO Mapping	Note
u32	rw		
Default Value	Minimum	Maximum	Unit
Description			
Reserved, do not use.			

8.12 Current Regulation

Address	Name		Mnemonic
41381	*Mode_CRRG*		
Data Type	Access Type	PD Mapping	Note
u8	rw	RPD	
Default Value	Minimum	Maximum	Unit
0			

Description

It defines how the current is supplied to the motor.

When the static mode is selected, the motor receives the *CurrentMax_MTRCNF* current while in movement, and the *CurrentMin_MTRCNF* current when it is stopped (after the time defined by the *CurrentIdleDelay_MTRCNF* register).

Instead, when the dynamic mode is selected, the drive supplies the motor with a current value proportional to the load. The current variation always occurs between the minimum and maximum values defined by the *CurrentMin_MTRCNF* and *CurrentMax_MTRCNF* registers respectively.

Value	Description
0	Static current supply independent from the load
1	Dynamic current supply proportional to the load

Address	Name			Mnemonic
41382	*Gain_CRRG*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u8	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	100	50	200	% (Ex. 120 = 120%)

Description

It allows to intervene manually on the open-loop gain.

The drive automatically determines the best parameter for the phase current regulation, nevertheless through this register it is possible to manually intervene on the gain up to halve it (50%) or double it (200%).

8.13 Motion Setup

Address	Name			Mnemonic
41411	*Feedback_MTNSTP*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u8	rw	RPD	It cannot be modified with the drive enabled.
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			

Description

The register is useful to enable and configure the closed loop operation of the motor.

The following table resumes the use of the register bits.

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>1..0</td><td>Feedback da Encoder
<table><tr><th>Value</th><th>Description</th></tr><tr><td>0</td><td>Disabled</td></tr><tr><td>1</td><td>Closed loop (feedback enabled)</td></tr><tr><td>2</td><td>Following control</td></tr></table>
It allows to enable the closed loop operation using the motor encoder (incremental encoder) for feedback.

Q When the following error control is selected, it is possible to use both the incremental encoder and the SSI encoder as position source.</td></tr>
<tr><td>2</td><td>Not used, set to 0.</td></tr>
<tr><td>3</td><td>Encoder phasing

It allows to choose the Encoder phasing mode.</td></tr>
</table>

	Value / Description: 0 – Only when necessary (ex. at the first motor enabling after the device power on) 1 – Always at each motor enabling
4	Feedback source Value / Description: 0 – Incremental encoder 1 – SSI Encoder
7..5	Not used, set to 0.

For a correct management of the closed loop besides activating the Encoder feedback, you must also configure the features of the encoder connected to the motor, through the *Configuration_ENCMTR* and *CPR_ENCMTR* registers.

In the *Operational* and *Quick Stop* status, it is not possible to modify the register value. The *Feedback_MTNSTP* register can be modified only with motor disabled.

Address	Name			Mnemonic
41412	*CurrentEnableRamp_MTNSTP*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		🖫
	Default Value	**Minimum**	**Maximum**	**Unit**
	10		2000	ms (ex. 250 = 250ms).

Description

Through this register it is possible to set the time required by the current to reach the rated value each time the motor is enabled.

A long current ramp can be of help to limit the peak current on the power supply and to damp the rotor alignment movement the first time the motor is enabled.

8.14 Position Loop

Address	Name			Mnemonic
41441	*Kp_PSTNLP*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	🖫
	Default Value	**Minimum**	**Maximum**	**Unit**
	300	2	3000	

Description

It allows to set the proportional gain of the position control loop.

	Too low a value causes a great position error while an excessive value can make the system unstable.

8.15 Error Class Setup

Address	Name			Mnemonic
41481	*Following_Error_ERRCS*			
	Data Type	Access Type	PD Mapping	Note
	u8	rw		🖫
	Default Value	Minimum	Maximum	Unit
	6			

Description

It allows you to define the emergency class generated in case of following error.
The following are the values that can be used:

Class	Description	Drive's reaction
2	Error that requires the stop of the motor but not the switch to FAULT.	Deceleration with Quick Stop.
4	Error that requires the stop of the motor and subsequently the switching to FAULT.	Deceleration with Quick Stop and subsequent disabling of the motor.
6	Error that requires the immediate disabling of the motor and the switching to FAULT.	Disabling of the motor.
8	As per class 6. The error can be reset only by a shutdown and restart cycle.	Disabling of the motor.

8.16 Digital Inputs Assignment

The registers described below allow you to assign functions and actions to the digital inputs.

For example, a digital input can be used to reset the fault as an alternative to the bit *FaultAcknowledge* contained in the *ControlDWord* register. Please note that the same input can be associated with more functions and actions.

Address	Name			Mnemonic
41812	*FaultReset_DIA*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		🖫
	Default Value	Minimum	Maximum	Unit
	0502_h			

Description

It allows to use a digital input to reset a fault condition.

The register has a dimension of 2 bytes, the low byte is used to select the input while the high one to specify the status to be considered active. For example, by selecting *Falling Edge* the fault will reset on the active / inactive transition (falling edge) of the input.

The following tables show the possible values that can be assigned to the low and high bytes.

High byte, active status	Low byte, input number

Value	Description
4	Rising Edge
5	Falling Edge

Value	Description
2	Digital input 2
3	Digital input 3
4	Digital input 4
5	Digital input 5
6	Digital input 6
7	Digital input 7

Address	Name			Mnemonic
41813	*PositiveLimit_DIA*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		🖫
	Default Value	**Minimum**	**Maximum**	**Unit**
	0207_h			

Description

It defines the input to which the positive limit switch is connected.

The register has a dimension of 2 bytes, the low byte is used to select the digital input while the high one to specify the status to be considered active. For example, by selecting *Inactive Input* the limit switch will be considered active when the input is switched off (inactive).

The following table shows the possible values that can be assigned to the high and low bytes.

High byte, active status	Low byte, input number

Value	Description
0	Always Inactive
1	Always Active
2	Active Input
3	Inactive Input

Value	Description
2	Digital input 2
3	Digital input 3
4	Digital input 4
5	Digital input 5
6	Digital input 6
7	Digital input 7

Address	Name			Mnemonic
41814	*NegativeLimit_DIA*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		🖫
	Default Value	Minimum	Maximum	Unit
	0206_h			

Description

It defines the input to which the negative limit switch is connected.

The register has a dimension of 2 bytes, the low byte is used to select the digital input while the high one to specify the status to be considered active. For example, by selecting *Inactive Input* the limit switch will be considered active when the input is switched off (inactive).

The following table shows the possible values that can be assigned to the high and low bytes.

High byte, active status

Value	Description
0	Always Inactive
1	Always Active
2	Active Input
3	Inactive Input

low byte, input number

Value	Description
2	Digital Input 2
3	Digital Input 3
4	Digital Input 4
5	Digital Input 5
6	Digital Input 6
7	Digital Input 7

Address	Name			Mnemonic
41815	*Home_DIA*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw		🖫
	Default Value	Minimum	Maximum	Unit
	0205_h			

Description

It defines the input to which the home switch is connected.

The register has a dimension of 2 bytes, the low byte is used to select the digital input while the high one to specify the status to be considered active. For example, by selecting *Inactive Input* the home switch will be considered active when the input is switched off (inactive).

The following table shows the possible values that can be assigned to the high and low bytes.

High byte, active status

Value	Description
0	Always Inactive
1	Always Active
2	Active Input
3	Inactive Input

low byte, input number

Value	Description
2	Digital input 2
3	Digital input 3
4	Digital input 4
5	Digital input 5
6	Digital input 6
7	Digital input 7

Address	Name	Mnemonic
41820	*Holding_Brake_DIA*	

Data Type	Access Type	PD Mapping	Note
u16	rw		🖫

Default Value	Minimum	Maximum	Unit
0			

Description

It defines the input used for the manual control of the holding brake.

The register has a size of 2 bytes, the low byte is used to select the digital input while the high one is used to specify the state to be considered active. For example, by selecting *inactive input* the brake will be commanded when the input is off (inactive).

The following table shows the possible values that can be assigned to the high and low byte.

High byte, active status

Value	Description
0	Always Inactive
1	Always Active
2	Active Input
3	Inactive Input

low byte, input number

Value	Description
2	Digital input 2
3	Digital input 3
4	Digital input 4
5	Digital input 5
6	Digital input 6
7	Digital input 7

If you do not need to manually control the holding brake, set the register to 0. For example, if you want to manually control the brake with input 7 active, set the register to 0207_h.

8.17 Digital Outputs Assignment

Address	Name	Mnemonic
41831	*Source_DOA_0*	

Data Type	Access Type	PD Mapping	Note
u16	rw		🖫

Default Value	Minimum	Maximum	Unit
8040_h			

Description

It allows to select the source to be assigned to the digital output 0 (DO0 signal).

Value	Description
0000_h	General Purpose
0008_h	Active
8008_h	Inactive
0040_h	Fault
8040_h	No Fault
0041_h	Operational enabled

8041$_h$	No Operational enabled
0042$_h$	Quick stop active
8042$_h$	No Quick stop active
0044$_h$	Holding Brake
8044$_h$	No Holding Brake
0050$_h$	Positive Movement
8050$_h$	No Positive Movement
0051$_h$	Negative Movement
8051$_h$	No Negative Movement
0060$_h$...006F$_h$	*Statusword* bit n = 1 (0060$_h$=bit0, 0061$_h$=bit1, etc.)
8060$_h$...806F$_h$	*Statusword* bit n = 0 (8060$_h$=bit0, 8061$_h$=bit1, etc.)

For example, by setting the value 8040$_h$, the output will be activated if there are no fault conditions. Instead, by setting the value 8067$_h$, the output will be activated every time the bit 7 (bit *warning*) of the *Statusword* will have value 0.

When the value 0000$_h$ (General Purpose) is selected the output status is controlled by the value of bit 0 of the *Outputs_DOV* register.

Address	Name			Mnemonic
41841	*Source_DOA_1*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0041$_h$			

Description

It allows to select the source to be assigned to the digital output 1 (DO1 signal).

Value	Description
0000$_h$	General Purpose
0008$_h$	Active
8008$_h$	Inactive
0040$_h$	Fault
8040$_h$	No Fault
0041$_h$	Operational enabled
8041$_h$	No Operational enabled
0042$_h$	Quick stop active
8042$_h$	No Quick stop active
0044$_h$	Holding Brake
8044$_h$	No Holding Brake
0050$_h$	Positive Movement
8050$_h$	No Positive Movement
0051$_h$	Negative Movement
8051$_h$	No Negative Movement
0060$_h$...006F$_h$	*Statusword* bit n = 1 (0060$_h$=bit0, 0061$_h$=bit1, etc.)
8060$_h$...806F$_h$	*Statusword* bit n = 0 (8060$_h$=bit0, 8061$_h$=bit1, etc.)

For example, by setting the value 8040$_h$, the output will be activated if there are no fault conditions. Instead, by setting the value 8067$_h$, the output will be activated every time the bit 7 (bit *warning*) of the *Statusword* will have value 0.

	When the value 0000_h (General Purpose) is selected, the output status is controlled by the value of the bit 1 value of the *Outputs_DOV* register.

Address	Name			Mnemonic
41851	*Source_DOA_2*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	$006A_h$			

Description

It allows to select the source to be assigned to the digital output 2 (DO2 signal).

Value	Description
0000_h	General Purpose
0008_h	Active
8008_h	Inactive
0040_h	Fault
8040_h	No Fault
0041_h	Operational enabled
8041_h	No Operational enabled
0042_h	Quick stop active
8042_h	No Quick stop active
0044_h	Holding Brake
8044_h	No Holding Brake
0050_h	Positive Movement
8050_h	No Positive Movement
0051_h	Negative Movement
8051_h	No Negative Movement
0060_h…$006F_h$	*Statusword* bit n = 1 (0060_h=bit0, 0061_h=bit1, etc.)
8060_h…$806F_h$	*Statusword* bit n = 0 (8060_h=bit0, 8061_h=bit1, etc.)

For example, by setting the value 8040_h, the output will be activated if there are no fault conditions. Instead, by setting the value 8067_h, the output will be activated every time the bit 7 (bit *warning*) of the *Statusword* will have value 0.

When the value 0000_h (General Purpose) is selected, the output status is controlled by the value of the bit 2 value of the *Outputs_DOV* register.

8.18 Device Status

Address	Name			Mnemonic
42141	*BridgeTemperature_DVSTS*			
	Data Type	Access Type	PD Mapping	Note
	i16	ro	TPD	
	Default Value	Minimum	Maximum	Unit
				0.1°C (Ex. 528 = 52.8°C)
	Description			
	It indicates the temperature reached by the power stage.			

Address	Name			Mnemonic
42143	*Power_Voltage_DVSTS*			
	Data Type	Access Type	PDO Mapping	Note
	u16	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
				0.1Vdc (Ex. 482 = 48.2Vdc)
	Description			
	It indicates the voltage of the power bus.			

8.19 Digital Inputs Value

Address	Name			Mnemonic
42820	*Inputs_DIV*			
	Data Type	Access Type	PD Mapping	Note
	u16	ro	TPD	
	Default Value	Minimum	Maximum	Unit
	Description			

It indicates the logical status of the digital inputs.

The bits of the register are associated to the digital inputs as follows:

Bit	Description
0	Digital Input 0 (DI0)
1	Digital Input 1 (DI1)
2	Digital Input 2 (DI2)
3	Digital Input 3 (DI3)
4	Digital Input 4 (DI4)
5	Digital Input 5 (DI5)
6	Digital Input 6 (DI6)
7	Digital Input 7 (DI7)
8..15	Reserved, ignore the value

A bit value = 1 indicates active input, on the contrary if the bit value is 0 it means that the logic status of the input is inactive.

8.20 Digital Outputs Value

Address	Name			Mnemonic
42922	*Outputs_DOV*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	🖫
	Default Value	**Minimum**	**Maximum**	**Unit**

Description

It indicates the logic status of the digital outputs and allows to set the value for the outputs configured as *General Purpose.*

The bits of the register are assigned to the digital outputs as follows:

Bit	Description
0	Digital Output 0 (DO0)
1	Digital Output 1 (DO1)
2	Digital Output 2 (DO2)
3..15	Reserved, ignore the value

A bit value = 1 indicates active output, on the contrary if the bit value is 0 it means that the logic status of the output is inactive.

If the output is assigned to a drive internal source (through the *Digital Output n Action* register) the modification of the corresponding bit will not be possible and any attempt of that kind will be ignored.

Address	Name			Mnemonic
42232	*SetOutput_DOV*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	U8	wo		
	Default Value	**Minimum**	**Maximum**	**Unit**

Description

It allows to modify one single bit of the *Outputs_DOV* register.

To modify one single bit of the *Outputs_DOV* register simply write this register with a value made up as follows: the bits from bit 0 to bit 3 must contain the number of bits of the *Outputs_DOV* register on which you want to act, while the bit 7 will indicate whether the bit must be set to 1 or 0. When the bit 7 is equal to 1, the bit of the *Outputs_DOV* register is put to 0 and vice versa.

For example, if the *SetOutput_DOV* register is written with the value 81_h, the bit 1 of the *Outputs_DOV* register will be set to 0, instead writing the value 01_h the same bit will be set to 1. If we write the *SetOutput_DOV* register with value 00_h the bit 0 of the *Outputs_DOV* register will be set to 1.

The following table shows the use of the bits of the register *SetOutput_DOV*.

Bit	Description
0..3	Number of the bits of the *Outputs_DOV* register on which you want to act. Value | Description 0 | Bit 0 1 | Bit 1 … | Bit n 15 | Bit 15
4..6	Not used, set to 0
7	Logic level to set in the bit of the *Outputs_DOV* register Value | Description 0 | Bit = 1 1 | Bit = 0

If the selected bit of the *Outputs_DOV* register corresponds to an output assigned to a drive internal source (through the *Digital Output n Action* register), the modification of the corresponding bit will not be possible and any attempt of that kind will be ignored.

8.21 Analog Inputs Value

Address	Name			Mnemonic
42251	*Voltage_AIV_0*			
	Data Type	Access Type	PD Mapping	Note
	i16	ro	TPD	
	Default Value	Minimum	Maximum	Unit
				1mV (Ec. 5302 = 5.302V)
	Description			
	It indicates the voltage at the analog input 0.			

Address	Name			Mnemonic
42261	*Voltage_AIV_1*			
	Data Type	Access Type	PD Mapping	Note
	i16	ro	TPD	
	Default Value	Minimum	Maximum	Unit
				1mV (Ex. 5302 = 5.302V)
	Description			
	It indicates the voltage at the analog input 1.			

8.22 Analog Outputs Value

Address	Name			Mnemonic
42271	*Voltage_AOV_0*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	i16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			1mV (Ex. 2450 = 2.450V)
	Description			
	It allows to set the voltage of the analog output 0.			

Address	Name			Mnemonic
42281	*Voltage_AOV_1*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	i16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			1mV (Ex. 2450 = 2.450V)
	Description			
	It allows to set the voltage of the analog output 1.			

8.23 Motor Configuration

Address	Name			Mnemonic
42312	*CurrentMin_MTRCNF*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	4000			0.01% (Ex. 2508 = 25.08%)
	Description			
	It allows to set the minimum value of the motor phase current. When the current regulation is set dynamically (through the *Mode_CRRG* register) the minimum current value represents the current supplied to the motor without load. Instead, if the current regulation is static, it defines then the idle current supplied to the motor after the *CurrentIdleDelay_MTRCNF* time from the stop. The value is expressed as a percentage of the motor rated current set by the *RatedCurrent_MTRDT* register. For example, if the motor has a rated current of 4Arms and a minimum current equal to 25%, the drive will never supply less than 1Arms to the motor The current value can be limited by the *TargetTorque* register.			

Address	Name			Mnemonic
42313	*CurrentMax_MTRCNF*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	8000			0.01% (Ex. 7550 = 75.5%)
	Description			
	It allows to set the maximum value of the motor phase current.			

	When the current regulation is set dynamically (through the *Mode_CRRG* register) the maximum current value represents the current supplied to the motor in locked rotor condition. Instead, if the current regulation is static, the *CurrentMax_MTRCNF* register defines the current supplied to the motor when it is rotating. The value is expressed as a percentage of the motor rated current set by the *RatedCurrent_MTRDT* register. For example, if the motor has a rated current of 4Arms and a minimum current equal to 75%, the drive will never supply less than 3Arms to the motor. The current value can be limited by the *TargetTorque* register.

Address	Name			Mnemonic
42314	*CurrentIdleDelay_MTRCNF*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw		
	Default Value	**Minimum**	**Maximum**	**Unit**
	500	2	10000	1ms (Ex. 3500 = 3.5s)
	Description			
	It allows to set the motor halting time before the current reaches the value defined by the *CurrentMin_MTRCNF* register. When the dynamic current regulation mode is active this register has no effect.			

8.24 Motor Value

Address	Name			Mnemonic
42320	*Current_MTRV*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	I16	ro	TPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
				10mArms (Ex. 225 = 2.25 Arms)
	Description			
	It indicates the actual current supplied to the motor expressed in effective value.			

Address	Name			Mnemonic
42324	*Utilization_MTRV*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	i16	ro	TPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
				0,1% (Ex. 705 = 70.5%)
	Description			
	It indicates the ratio between the torque used by the load and the actual torque that can be supplied by the motor.			

8.25 Motor Incremental Encoder Value

Address	Name			Mnemonic
42331	*Status_ENCMTRV*			
	Data Type	Access Type	PDO Mapping	Note
	u8	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
	Description			
	The register is useful to know the status of the motor incremental encoder. The following table shows the meaning of the bits of the register.			

Bit	Description
0	Logic level of the Index signal
1	Logic level of the phase B
2	Logic level of the phase A
7..3	Not used. Ignore the value.

Address	Name			Mnemonic
42332	*PositionRaw_ENCMTRV*			
	Data Type	Access Type	PDO Mapping	Note
	u32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
	Description			
	It indicates the cyclical position on the revolution of the motor incremental encoder.			

Address	Name			Mnemonic
42334	*Frequency_ENCMTRV*			
	Data Type	Access Type	PDO Mapping	Note
	u32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
				Hz (Ex. 4365 = 4.365KHz)
	Description			
	It indicates the frequency of the pulses generated by the motor incremental encoder.			

8.26 SSI Encoder Value

Address	Name			Mnemonic
42339	*Status_ENCSSIV*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	u8	ro	TPDO	
	Default Value	**Minimum**	**Maximum**	**Unit**
				10mArms (Ex. 225 = 2.25 Arms)

Description

The register is useful to know the status of the SSI encoder.

The following table shows the meaning of the bits of the register.

<table>
<tr><th>Bit</th><th>Description</th></tr>
<tr><td>0</td><td>Valid data
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>SSI encoder reading not operative or with errors</td></tr>
<tr><td>1</td><td>SSi encoder reading operative</td></tr>
</table>
When the bit value is 0, the Position_ENCSSIV and PositionRaw_ENCSSIV registers may non reflect the position of the SSI encoder.</td></tr>
<tr><td>1</td><td>Errors
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>No errors in the data frame</td></tr>
<tr><td>1</td><td>Errors in the data frame</td></tr>
</table></td></tr>
<tr><td>2</td><td>Not used. Ignore the value.</td></tr>
<tr><td>3</td><td>Power supplied
<table>
<tr><th>Value</th><th>Description</th></tr>
<tr><td>0</td><td>No encoder power supply</td></tr>
<tr><td>1</td><td>Encoder power supply</td></tr>
</table>
The bit assumes significance when the SSI encoder is configured with the bit3 of the Configuration_ENCSSI register to 1, differently the bit is set to 0.</td></tr>
<tr><td>7..4</td><td>Not used. Ignore the value.</td></tr>
</table>

Address	Name			Mnemonic
42340	*Position_ENCSSIV*			
	Data Type	Access Type	PDO Mapping	Note
	i32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
				0.0001rev (Ex. 5000 = 0.5000rev)
	Description			
	It indicates the position expressed in 0.0001rev obtained from the SSI encoder. The formula used to obtain the position expressed in 0.0001rev is the following: *Position_ENCSSIV* = (*PositionRaw_ENCSSIV* - *PositionOffset_ENCSSI*) * *ScalingFactor_N_ENCSSI* / *ScalingFactor_D_ENCSSI*			

Address	Name			Mnemonic
42342	*PositionRaw_ENCSSIV*			
	Data Type	Access Type	PDO Mapping	Note
	u32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
	Description			
	It indicates the quota read by the SSI encoder.			

8.27 Holding Brake Control

Address	Name			Mnemonic
42361	*Control_HBRKC*			
	Data Type	Access Type	PDO Mapping	Note
	u8	rw	RPD	🖫
	Default Value	Minimum	Maximum	Unit
	0			
	Description			
	It allows you to manually control the holding brake. The following table shows the use of the bits of the register.			

Bit	Description
0	Manual Control It allows you to manually control the holding brake. Value 0: Inactive manual control signal Value 1: Active manual control signal he effect on the brake of the logic level assumed by this bit depends

Value	Description
0	Inactive manual control signal
1	Active manual control signal

	on the configuration of the *Option_HBRKS register,* therefore for a detailed description refer to chapter 8.11 Holding Brake Setup The bit0 is in logic OR with the digital input chosen for the manual control of the brake.
1..7	Not used. Set to 0.

Address	Name			Mnemonic
42362	*Status_HBRKC*			
	Data Type	Access Type	PDO Mapping	Note
	u8	ro	TPD	
	Default Value	Minimum	Maximum	Unit

Description

The register is useful to know the state of the holding brake.

The following table resumes the meaning of the bits of the register.

Bit	Description
0	Active It indicates the actual state of the output arranged for the control of the holding brake. Value | Description 0 | The brake control output is in the inactive state 1 | The brake control output is in the active state
1	Released It allows you to know if the *Release_Time_HBRKS* has elapsed and the brake is then in the released state. Value | Description 0 | The brake is not released 1 | The brake is released
2	Engaged

It allows you to know if the *Application_Time_HBRKS* has elapsed and the brake is then in the engaged state.

Value	Description
0	The brake is not engaged
1	The brake is engaged

8.28 Profile Torque Configuration

Address	Name			Mnemonic
42471	*Mode_PTCNF*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u8	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			

Description

It allows to configure the options of the *Torque* mode.

Normally in the torque control the motor speed is limited only by the load and by the characteristic of the motor. However, in many applications it is useful to set a maximum speed to prevent the motor to reach high speeds without load. Setting to 1 the bit 0 of the *Mode_PTCNF* register, the drive limits the motor maximum speed to the value set through the TargetV*elocity* register.

The following table shows the use of the bits of the register.

Bit	Description
0	Speed limitation Value 0: Speed not limited by the *TargetVelocity* register Value 1: Speed limited by the *TargetVelocity* register
1..7	Not used. Set to 0.

8.29 StatusDWord and ControlDWord

Address	Name			Mnemonic
42811	*StatusDWord*			
	Data Type	Access Type	PD Mapping	Note
	u16	ro	TPD	
	Default Value	Minimum	Maximum	Unit

Description

It allows you to know the device status.

The register is structured in bits whose meaning is described in the following table.

StatusDWord

bit	Description	bit	Description
31	Reserved, ignore the value	15	PositionReached, 1=true
30	Reserved, ignore the value	14	VelocityReached, 1=true
29	Reserved, ignore the value	13	TorqueReached, 1=true
28	Reserved, ignore the value	12	Standstill, 1=true
27	Reserved, ignore the value	11	Reserved, ignore the value
26	OperationMode	10	ReferenceSet, 1=true
25		9	TriggerAcknowledge
24	<table><tr><th>bit 26..24</th><th>Operation Mode</th></tr><tr><td>001</td><td>Position</td></tr><tr><td>010</td><td>Velocity</td></tr><tr><td>011</td><td>Torque</td></tr><tr><td>111</td><td>Homing</td></tr></table>The bit combination not in the table must be considered as non-selected operating mode.	8	Busy, 1= operation in progress
23	Reserved, ignore the value	7	Reserved, ignore the value
22	Reserved, ignore the value	6	Reserved, ignore the value
21	Reserved, ignore the value	5	Warning, 1= active
20	Reserved, ignore the value	4	Fault, 1=active
19	PositionError, 1=true	3	Halt, 1= active
18	VelocityError, 1=true	2	QuickStop, 1=active
17	Reserved, ignore the value	1	Enable
16	Reserved, ignore the value	0	<table><tr><th>bit 1</th><th>bit 0</th><th>Description</th></tr><tr><td>0</td><td>0</td><td>Disabled</td></tr><tr><td>0</td><td>1</td><td></td></tr><tr><td>1</td><td>0</td><td></td></tr><tr><td>1</td><td>1</td><td>Enabled</td></tr></table>

The reserved bits must be ignored. Whichever their value, the application must not modify its behavior.

Address	Name			Mnemonic
42911	*ControlDWord*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	
	Default Value	Minimum	Maximum	Unit

Description

It allows you to control the device.

The register is structured in bits whose functions are described in the following table.

<table>
<tr><th colspan="4">ControlDWord</th></tr>
<tr><th>bit</th><th>Description</th><th>Bit</th><th>Description</th></tr>
<tr><td>31</td><td>Reserved, set to 0</td><td>15</td><td>Reserved, set to 0</td></tr>
<tr><td>30</td><td>Reserved, set to 0</td><td>14</td><td>Reserved, set to 0</td></tr>
<tr><td>29</td><td>Reserved, set to 0</td><td>13</td><td>Reserved, set to 0</td></tr>
<tr><td>28</td><td>Reserved, set to 0</td><td>12</td><td>Reserved, set to 0</td></tr>
<tr><td>27</td><td>Reserved, set to 0</td><td>11</td><td>Reserved, set to 0</td></tr>
<tr><td>26</td><td rowspan="3">OperationMode
<table>
<tr><th>bit 26..24</th><th>Operation Mode</th></tr>
<tr><td>000</td><td>None</td></tr>
<tr><td>001</td><td>Position</td></tr>
<tr><td>010</td><td>Velocity</td></tr>
<tr><td>011</td><td>Torque</td></tr>
<tr><td>111</td><td>Homing</td></tr>
</table>
The bits combination not in the table must be avoided.</td><td>10</td><td>AbsoluteRelative, 1=Relative</td></tr>
<tr><td>25</td><td>9</td><td>Trigger, ↑=trigger</td></tr>
<tr><td>24</td><td>8</td><td>TriggerMode, 1=Active</td></tr>
<tr><td>23</td><td>Reserved, set to 0</td><td>7</td><td>Reserved, set to 0</td></tr>
<tr><td>22</td><td>Reserved, set to 0</td><td>6</td><td>Reserved, set to 0</td></tr>
<tr><td>21</td><td>Reserved, set to 0</td><td>5</td><td>WarningAcknowledge, ↑=accepted</td></tr>
<tr><td>20</td><td>Reserved, set to 0</td><td>4</td><td>FaultAcknowledge, ↑=accepted</td></tr>
<tr><td>19</td><td>PositionErrorLatch, 1=Latch</td><td>3</td><td>Halt, 1= active</td></tr>
<tr><td>18</td><td>VelocityErrorLatch, 1=Latch</td><td>2</td><td>QuickStop, 1=active</td></tr>
<tr><td>17</td><td>Reserved, set to 0</td><td>1</td><td rowspan="2">Enable
<table>
<tr><th>bit 1</th><th>bit 0</th><th>Description</th></tr>
<tr><td>0</td><td>0</td><td rowspan="3">Disabled</td></tr>
<tr><td>0</td><td>1</td></tr>
<tr><td>1</td><td>0</td></tr>
<tr><td>1</td><td>1</td><td>Enabled</td></tr>
</table></td></tr>
<tr><td>16</td><td>Reserved, set to 0</td><td>0</td></tr>
</table>

The reserved bits must be set to 0 to ensure compatibility with future firmware revisions.

8.30 Errori

Address	Name			Mnemonic
42711	*ErrorCode*			
	Data Type	Access Type	PD Mapping	Note
	u16	ro	TPD	
	Default Value	Minimum	Maximum	Unit
	Description			
	It provides the code of the last error occurred in the device.			

Address	Name			Mnemonic
42713	*ErrorSubCode*			
	Data Type	Access Type	PD Mapping	Note
	U32	ro	TPD	
	Default Value	Minimum	Maximum	Unit
	Description			
	It contains additional information on the last error occurred in the device.			

8.31 Actual Motion Values

Address	Name			Mnemonic
42813	*ActualTorque*			
	Data Type	Access Type	PD Mapping	Note
	i16	ro	TPD	
	Default Value	Minimum	Maximum	Unit
				0.1% (Ex. 405 = 40.5%)
	Description			
	It provides the actual value of the torque. The value is given in 0.1% of the rated motor torque.			

Address	Name			Mnemonic
42814	*ActualVelocity*			
	Data Type	Access Type	PD Mapping	Note
	i32	ro	TPD	
	Default Value	Minimum	Maximum	Unit
				0.1rpm (Ex. 3850 = 385.0rpm)
	Description			
	The register provides the actual velocity value. The value is given in 0.1rpm			

Address	Name			Mnemonic
42816	*ActualPosition*			
	Data Type	Access Type	PD Mapping	Note
	i32	ro	TPD	
	Default Value	Minimum	Maximum	Unit
				0.0001rev (Ex. 5000 = 0.5000rev)
	Description			
	The register provides the actual position value. The value is given in 1/10000 of revolution.			

Address	Name			Mnemonic
42818	*ActualPositionError*			
	Data Type	Access Type	PD Mapping	Note
	i32	ro	TPD	
	Default Value	Minimum	Maximum	Unit
				0,0001rev (Ex. 20000 = 2.0000rev)
	Description			
	This register allows you to read the following position error value. The value is given in 1/10000 of revolution.			

8.32 Target Motion Values

Address	Name			Mnemonic
42913	*TargetTorque*			
	Data Type	Access Type	PD Mapping	Note
	i16	Rw	RPD	💾
	Default Value	Minimum	Maximum	Unit
	0	-1300	1300	0.1% (Ex. 405 = 40.5%)
	Description			
	This register allows you to set the wished torque value (target torque) in the *Torque* mode. In the *Velocity* or *Position* mode it sets the maximum torque value of the motor, if the operation is in closed-loop mode. In open-loop, *TargetTorque* acts by limiting the phase current to the motor and is usually set to value 1000, corresponding to the 100% of the current. In open-loop mode the motor current is typically set using the *CurrentMin_MTRCNF* and *CurrentMax_MTRCNF* registers. The value is given in 0.1% of the rated motor torque.			

Address	Name			Mnemonic
42914	*TargetVelocity*			
	Data Type	Access Type	PD Mapping	Note
	i32	rw	RPD	💾
	Default Value	Minimum	Maximum	Unit
	0			0.1rpm (Ex. 1000 = 1000rpm)
	Description			
	The register allows you to set the target velocity in the *Velocity* mode. In the *Position* mode, it defines the full speed reached at the end of the acceleration ramp.			

	The value is given in 0.1rpm..

Address	Name			Mnemonic
42916	*TargetPosition*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	i32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			0.0001rev (Ex. 50000 = 5.0000rev)
	Description			
	It allows you to set the motor target position in the *Position* mode. The dives interprets the value of the *TargetPosition* register in absolute or relative mode according to the value of the *AbsoluteRelative* bit set in the *ControlDWord* register. The value is given in 1/10000 of revolution.			

Address	Name			Mnemonic
42918	*Acceleration*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	100	5	300000	rpm/s (Ex. 100 = 100rpm/s)
	Description			
	It allows you to set the acceleration value used during the execution of a motion profile. The value is given in rpm/s.			

Address	Name			Mnemonic
42920	*Deceleration*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	300	5	300000	rpm/s (Ex. 300 = 300rpm/s)
	Description			
	It allows you to set the deceleration value used during the execution of a motion profile. The value is given in rpm/s.			

Address	Name			Mnemonic
42521	*QuickStopDeceleration*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	5000	5	300000	rpm/s (Ex. 5000 = 5000rpm/s)
	Description			
	It allows you to set the deceleration value used during the emergency *Quick stop*. The value is given in rpm/s.			

Address	Name			Mnemonic
42523	*TorqueSlope*			
	Data Type	Access Type	PD Mapping	Note
	u32	rw	RPD	🖫
	Default Value	Minimum	Maximum	Unit
				0.1%/s (Ex. 558 = 55.8%/s)
	Description			
	This register allows to set the spped with which the torque available on the motor varies in the *Torque* mode. The value is given in thousandths of the rated motor torque per second.			

8.33 Limits

Address	Name			Mnemonic
42525	*MaxTorque*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	🖫
	Default Value	Minimum	Maximum	Unit
	0	-1300	1300	0.1% (Ex. 405 = 40.5%)
	Description			
	The register allows you to set the maximum torque value that can be supplied by the motor when the drive operates in closed-loop. In open-loop, *MaxTorque* acts by limiting the phase current to the motor and is usually set to value 1000, corresponding to the 100% of the current. In fact, in the open-loop mode the motor current is typically set using the *CurrentMin_MTRCNF* and *CurrentMax_MTRCNF* registers. The value is given in the 0.1% of the rated motor torque.			

Address	Name			Mnemonic
42526	*MaxSpeed*			
	Data Type	Access Type	PD Mapping	Note
	u32	rw	RPD	🖫
	Default Value	Minimum	Maximum	Unit
	30000	1	30000	0.1rpm (Ex. 5000 = 500.0rpm)
	Description			
	This register allows you to set the maximum speed allowed during a motion profile in both directions of rotation. The value is given in 0.1rpm.			

Address	Name			Mnemonic
42528	*MinPosition*			
	Data Type	Access Type	PD Mapping	Note
	i32	rw	RPD	🖫
	Default Value	Minimum	Maximum	Unit
	0			0.0001rev (Ex. -1000000 = -100rev)
	Descriptionn			
	This register allows you to set the minimum position software limit.			

The value is given in in 1/10000 of revolution.

To disable the position software limits the *MinPosition* and *MaxPosition* registers must both be set to 0.

Address	Name			Mnemonic
42530	*MaxPosition*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	i32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0			0.0001rev (Ex. 1000000 = 100rev)

Description

This register allows you to set the maximum position software limit.

The value is given in 1/10000 of revolution.

To disable the position software limits the *MinPosition* and *MaxPosition* registers must be both set to 0.

8.34 Validity and Error Windows

Address	Name			Mnemonic
42541	*WindowTorque*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	100			0.1rpm (Ex. 1000 = 100.0rpm)

Description

This register allows you to set the torque acceptance range with respect to the set target.

When the actual motor torque differs, with respect to the target value, less (in absolute value) than the value set using the *WindowTorque* register*,* the target speed is considered reached after the *WindowTorqueTime* time and the bit13 *TorqueReached* of StatusDWord is set to 1.

The value is given in 0.1rpm.

Address	Name			Mnemonic
42542	*WindowTorqueTime*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0		30000	1ms (Ex. 200 = 200ms)

Description

It allows you to set the time taken to validate the reached torque condition (speed error contained inside *WindowTorque* in absolute value).

The value is given in ms.

Address	Name			Mnemonic
42543	*WindowVelocity*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	100			0.1rpm (Ex. 1000 = 100.0rpm)
	Description			
	This register allows you to set the speed acceptance range with respect to the set target. When the actual motor speed differs, with respect to the target value, less (in absolute value) than the value set using the *WindowVelocity* register, the target speed is considered reached after the *WindowVelocityTime* time and the bit14 *VelocityReached* in the *StatusDWord* is set to 1. The value is given in 0.1rpm.			

Address	Name			Mnemonic
42544	*WindowVelocityTime*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u16	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0		30000	1ms (Ex. 200 = 200ms)
	Description			
	It allows you to set time taken to validate the reached speed condition (speed error contained inside *WindowVelocity* in absolute value). The value is given in ms.			

Address	Name			Mnemonic
42545	*WindowVelocityError*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	i32	ro	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	1000	0	30000	0.1rpm (Ex. 1000 = 100.0rpm)
	Description			
	It allows you to set the maximum "slippage" between the actual motor speed and the required speed. Once this value is exceeded (in absolute value), after the *WindowVelocityErrorTimeOut* time is elapsed, the bit18 *VelocityError* in the *StatusDWord* is set to 1. If the value of the *WindowVelocityError* register is set to $FFFFFFFF_h$ the following error control is disabled. The drive is able to know the actual motor speed only if the motor is provided with encoder and the drive is configured in the closed-loop mode. The value is given in 0.1rpm.			

Address	Name			Mnemonic
42547	*WindowVelocityErrorTimeOut*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	10		30000	1ms (Ex. 280 = 280ms)
	Description			
	It allows you to set the time taken to validate the speed error condition (speed error greater than *WindowVelocityError* in absolute value). The drive is able to know the actual motor speed only if the motor is provided with encoder and the drive is configured in the closed-loop mode.			

Address	Name			Mnemonic
42548	*WindowZeroSpeed*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	60			0.1rpm (Ex. 500 = 50.0rpm)
	Description			
	This register allows you to set the speed value beyond which the motor is considered in motion. When the actual motor speed becomes lower (in absolute value) than the value set by the *WindowZeroSpeed* register, the motor is considered at stand still and the bit12 *Standstill* is set to 1. The value is given in 0.1rpm.			

Address	Name			Mnemonic
42549	WindowZeroSpeedTime			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	0		30000	1ms (Ex. 200 = 200ms)
	Description			
	It allows you to set the time taken to validate the motor running condition. The value is given in ms.			

Address	Name			Mnemonic
42550	*WindowPosition*			
	Data Type	Access Type	PD Mapping	Note
	u32	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	10			0.0001rev (Ex. 20000 = 2.0000rev)
	Description			
	The register allows you to set the position acceptance space with respect to the set target. When the actual motor position differs, with respect to the target value, less (in absolute value) than the value set by the *WindowPosition* register, the target position is considered reached after *WindowPositionTime* and the bit14 *VelocityReached of the StatusDWord* is set to 1.			

	The value is given in 1/10000 of revolution.

Address	Name			Mnemonic
42552	*WindowPositionTime*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	0		30000	1ms (Ex. 600 = 0.6s)
	Description			
	It allows you to set the time taken to validate the condition of position reached (position error contained inside *WindowPosition* in absolute value). The value is given in ms.			

Address	Name			Mnemonic
42553	*WindowPositionError*			
	Data Type	Access Type	PD Mapping	Note
	u32	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	10000			0.0001rev (Ex. 20000 = 2.0000rev)
	Description			
	This register allows you to set the symmetrical positive and negative tolerance space between the actual position and the position request made by the trajectory generator. If the current position remains external to this value (in absolute value) for a time longer than *WindowPositionErrorTimeOut*, the following error is activated and the bit19 *PositionError* of the *StatusDWord* is set to 1. If the value of the *WindowPositionError* register is set to $FFFFFFFF_h$ the control on the following error is disabled. The value is given in 1/10000 of revolution.			

Address	Name			Mnemonic
42555	*WindowPositionErrorTimeOut*			
	Data Type	Access Type	PD Mapping	Note
	u16	rw	RPD	
	Default Value	Minimum	Maximum	Unit
	10		30000	1ms (Ex. 250 = 250ms)
	Description			
	It allows you to set the time taken to validate the condition of position error (position error greater than *WindowPositionError* in absolute value). The value is given in ms.			

8.35 Homing

Address	Name			Mnemonic
42581	*HomingMethod*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	i8	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**

Description

This register allows you to select the homing method procedure.

The following table describes the available homing methods and the sensors used by each of them. The abbreviations used have the following meaning: **PLS**=positive limit switch, **NLS**=negative limit switch, **HS**=homing switch, **IDX**=index.

Homing methods selectable through the* HomingMethod *register

Code	Description	Sensors used			
		NLS	NLS	HS	IDX
-80	Homing on the position read by the SSI encoder				
1	At the start, if negative limit switch inactive counterclockwise direction up to the limit switch, then reverse and homing at the first index outside the negative limit switch. At the start, if negative limit switch active clockwise direction up to leave the limit switch, then homing at the first index outside the negative limit switch.	●			●
2	At the start, if positive limit switch inactive clockwise direction up to the limit switch, then reverse and homing at the first index outside the positive limit switch. At the start, if positive limit switch active counterclockwise direction up to leave the limit switch, then homing at the first index outside the positive limit switch.		●		●
3	At the start, if home switch inactive initial direction clockwise up to home switch, then reverse and homing at the first index outside the home switch. At the start, if home switch active initial direction counterclockwise up to leave the switch, then homing at the first index outside the home switch.			●	●
4	At the start, if home switch inactive initial direction clockwise up to home switch, then homing at the first index inside the home switch. At the start, if home switch active initial direction counterclockwise up to leave the switch, then reverse and homing at the first index inside the home switch.			●	●
5	At the start, if home switch active initial direction clockwise up to leave the switch, then homing at the first index outside the home switch. At the start, if home switch inactive initial direction counterclockwise up to find the switch, then reverse and homing at the first index outside the home switch.			●	●
6	At the start, if home switch active initial direction clockwise up to leave the switch, then reverse and homing at the first index inside the home switch.			●	●

		At the start, if home switch inactive initial direction counterclockwise up to find the switch, then reverse and homing at the first index inside the home switch.				
	7	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, reverse and homing at the first index outside the switch. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then homing at the first index outside the switch. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then homing at the first index outside the switch.	●		●	●
	8	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, homing at the first index inside the home switch. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then reverse up to find again the home switch and finally homing at the first index inside the switch. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then reverse and homing at the first index inside the switch.	●		●	●
	9	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, continue up to leave it then reverse and homing at the first index inside the switch. In case of positive limit switch, reverse up to find the home switch, then homing at the first index inside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the first index inside the home switch.	●		●	●
	10	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, continue up to leave it then homing at the first index outside the home switch. In case of positive limit switch, reverse up to find the home switch, then reverse up to leave the switch and homing at the first index outside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then homing at the first index outside the home switch.	●		●	●
	11	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, reverse and homing at the first index outside the home switch. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch and then homing at the first index outside the switch. At the start, if home switch active initial direction		●	●	●

		clockwise up to leave the home switch, then homing at the first index outside the home switch.				
	12	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, homing at the first index inside the home switch. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch and then reverse up to find again the home switch and finally homing at the first index inside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the first index inside the home switch.		●	●	●
	13	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then reverse and homing at the first index inside the switch. In case of negative limit switch, reverse up to find the home switch, then homing at the first index inside the switch. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the first index inside the home switch.		●	●	●
	14	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then homing at the first index outside the home switch. In case of negative limit switch, reverse up to find the home switch, reverse up to leave the switch and then homing at the first index outside the switch. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then homing at the first index outside the switch.		●	●	●
	17	At the start, if negative limit switch inactive initial direction counterclockwise up to find the limit switch, then reverse and homing at the active/inactive switch transition. At the start, if negative limit switch active initial direction clockwise with homing at the active/inactive switch transition.		●		
	18	At the start, if positive limit switch inactive initial direction clockwise up to the limit switch, then reverse and homing at the active/inactive switch transition. At the start, if negative limit switch active initial direction counterclockwise with homing at the active/inactive switch transition.	●			
	19	At the start, if home switch inactive initial direction clockwise up to the home switch, then reverse and homing at the active/inactive switch transition. At the start, if home switch active initial direction counterclockwise with homing at the active/inactive switch transition.			●	

	20	At the start, if home switch inactive initial direction clockwise up to the home switch, then homing at the inactive/active switch transition. At the start, if home switch active initial direction counterclockwise up to leave the switch, then reverse and homing at the inactive/active switch transition.			●	
	21	At the start, if home switch active initial direction clockwise up to leave the switch, then homing at the active/inactive switch transition. At the start, if home switch inactive initial direction counterclockwise up to find the switch, then reverse and homing at the active/inactive switch transition.			●	
	22	At the start, if home switch active initial direction clockwise up to leave the switch, then reverse and homing at the inactive/active switch transition. At the start, if home switch inactive initial direction counterclockwise up to find the switch, then homing at the inactive/active switch transition.			●	
	23	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, reverse and homing at the active/inactive switch transition. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then homing at the active/inactive switch transition. At the start, if home switch active initial direction counterclockwise up to leave the switch, then homing at the active/inactive switch transition.	●		●	
	24	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, homing at the inactive/active switch transition. In case of positive limit switch, reverse up to find the home switch, continue up to leave the home switch, then reverse up to find again the home switch and finally homing at the inactive/active switch transition. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.	●		●	
	25	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit switch. In case of home switch, continue up to leave it, then reverse and homing at the inactive/active switch transition. In case of positive limit switch, reverse up to find the home switch, homing at the inactive/active switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.	●		●	
	26	At the start, if home switch inactive initial direction clockwise up to find the home switch or the positive limit	●		●	

		switch. In case of home switch, continue up to leave it, then homing at the active/inactive switch transition. In case of positive limit switch, reverse up to find the home switch, then reverse up to leave the switch and finally homing at the active/inactive switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then homing at the active/inactive switch transition.				
	27	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, reverse and homing at the active/inactive switch transition. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch, then homing at the active/inactive switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then homing at the active/inactive switch transition.		●	●	
	28	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, homing at the inactive/active switch transition. In case of negative limit switch, reverse up to find the home switch, continue up to leave the home switch, then reverse up to find again the home switch and finally homing at the inactive/active switch transition. At the start, if home switch active initial direction clockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.		●	●	
	29	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then reverse and homing at the inactive/active switch transition. In case of negative limit switch, reverse up to find the home switch, then homing at the inactive/active switch transition. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then reverse and homing at the inactive/active switch transition.		●	●	
	30	At the start, if home switch inactive initial direction counterclockwise up to find the home switch or the negative limit switch. In case of home switch, continue up to leave it then homing at the active/inactive switch transition. In case of negative limit switch, reverse up to find the home switch, then reverse up to leave the switch, then homing at the active/inactive switch transition. At the start, if home switch active initial direction counterclockwise up to leave the home switch, then homing at the active/inactive switch transition.		●	●	
	33	Initial direction counterclockwise with homing at the first				●

	index found.				
34	Initial direction clockwise with homing at the first index found.				●
37	Homing at the actual position.				

Address	Name			Mnemonic
42582	*HomingSpeedForSwitch*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	600		30000	0.1rpm (Ex. 600 = 60.0rpm)

Description

It allows you to set the speed during search for switch in homing procedure.

The value is given in 0.1rpm.

Address	Name			Mnemonic
42584	*HomingSpeedForZero*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	u32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
	60		30000	0.1rpm (Ex. 100 = 10.0rpm)

Description

It allows you to set the speed during search for zero in homing procedure.

The value is given in 0.1rpm.

Address	Name			Mnemonic
42586	*HomingOffset*			
	Data Type	**Access Type**	**PD Mapping**	**Note**
	i32	rw	RPD	
	Default Value	**Minimum**	**Maximum**	**Unit**
				0.0001rev (Ex. 20000 = 2.0000rev)

Description

This register allows you to define the difference between the zero logic position of the application and the home position in the machine.

When the homing procedure ends successfully the *ActualPosition* register is set to the value of the *HomingOffset* register.

The value is given in 1/10000 of revolution.

8.36 Touch Probe function

Address	Name			Mnemonic
42591	*TouchProbeFunction*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	u16	rw	RPDO	
	Default Value	**Minimum**	**Maximum**	**Unit**
	0x0000			

Description

It allows you to configure the touch probe functions.
The bits of the lower byte define the *Touch probe 1* functions while the higher byte contains the bits related to *Touch probe 2*.

The following table shows the use of the bits and the allowed values:

Touch probe 2		*Touch probe 1*	
Bit	Description	Bit	Description
15, 14	Not used. Set to 0	7, 6	Not used. Set to 0
13	0 – Switch off sampling at negative edge 1 – Enable sampling at negative edge	5	0 – Switch off sampling at negative edge 1 – Enable sampling at negative edge
12	0 – Switch off sampling at positive edge 1 – Enable sampling at positive edge	4	0 – Switch off sampling at positive edge 1 – Enable sampling at positive edge
11, 10	00 – Trigger with touch probe 2 input 01 – Trigger with Encoder Index	3, 2	00 – Trigger with touch probe 1 input 01 – Trigger with Encoder Index
9	0 – Trigger first event	1	0 – Trigger first event
8	0 – Switch off touch probe 2 1 – Enable touch probe 2	0	0 – Switch off touch probe 1 1 – Enable touch probe 1

When the *Trigger with Encoder Index* mode is selected, also the sampling on the positive edge must be selected (*Enable sampling at positive edge*).

Address	Name			Mnemonic
42592	*TouchProbeStatus*			
	Data Type	**Access Type**	**PDO Mapping**	**Note**
	u16	ro	TPDO	
	Default Value	**Minimum**	**Maximum**	**Unit**

Description

It indicates the touch probe status.
The bits of the lower byte are related to the *Touch probe 1* while the bits in the higher byte are related to the *Touch probe 2*.

The following table shows the meaning of each bit:

Touch probe 2		*Touch probe 1*	
Bit	**Description**	**Bit**	**Description**
15, 14	Reserved bits. Ignore the value	7, 6	Reserved bits. Ignore the value
13..11	0: Reserved	5..3	0: Reserved
10	0: Touch probe 2 no negative edge value stored 1: Touch probe 2 negative edge position stored	2	0: Touch probe 1 no negative edge value stored 1: Touch probe 1 negative edge position stored
9	0: Touch probe 2 no positive edge value stored 1: Touch probe 2 positive edge position stored	1	0: Touch probe 1 no positive edge value stored 1: Touch probe 1 positive edge position stored
8	0: Touch probe 2 is switched off 1: Touch probe 2 is enabled	0	0: Touch probe 1 is switched off 1: Touch probe 1 is enabled

Address	Name			Mnemonic
42593	*TouchProbe_1_PositiveEdge*			
	Data Type	Access Type	PDO Mapping	Note
	i32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
				0.0001rev (Ex. 25000 = 2.5000rev)
	Description			
	It contains the position captured on the positive edge of the touch probe 1. The value is given in 1/10000 per revolution.			

Address	Name			Mnemonic
42595	*TouchProbe_1_NegativeEdge*			
	Data Type	Access Type	PDO Mapping	Note
	i32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
				0.0001rev (Ex. 25000 = 2.5000rev)
	Description			
	It contains the position captured on the negative edge of the touch probe 1. The value is given in 1/10000 per revolution.			

Address	Name			Mnemonic
42597	*TouchProbe_2_PositiveEdge*			
	Data Type	Access Type	PDO Mapping	Note
	i32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
				0.0001rev (Ex. 25000 = 2.5000rev)
	Description			
	It contains the position captured on the positive edge of the touch probe 2. The value is given in 1/10000 of revolution.			

Address	Name			Mnemonic
42599	*TouchProbe_2_NegativeEdge*			
	Data Type	Access Type	PDO Mapping	Note
	i32	ro	TPDO	
	Default Value	Minimum	Maximum	Unit
				0.0001rev (Ex. 25000 = 2.5000rev)
	Description			
	It contains the position captured on the negative edge of the touch probe 2. The value is given in 1/10000 of revolution.			

LAM Technologies
Viale Ludovico Ariosto, 492/D
50019 Sesto Fiorentino
Firenze - ITALY

Ph. +39 055 4207746
Fax +39 055 4207651

www.lamtechnologies.com

General information info@lamtechnologies.com
Technical support support@lamtechnologies.com
Sales information sales@lamtechnologies.com

rev. 0.02 (preliminary)