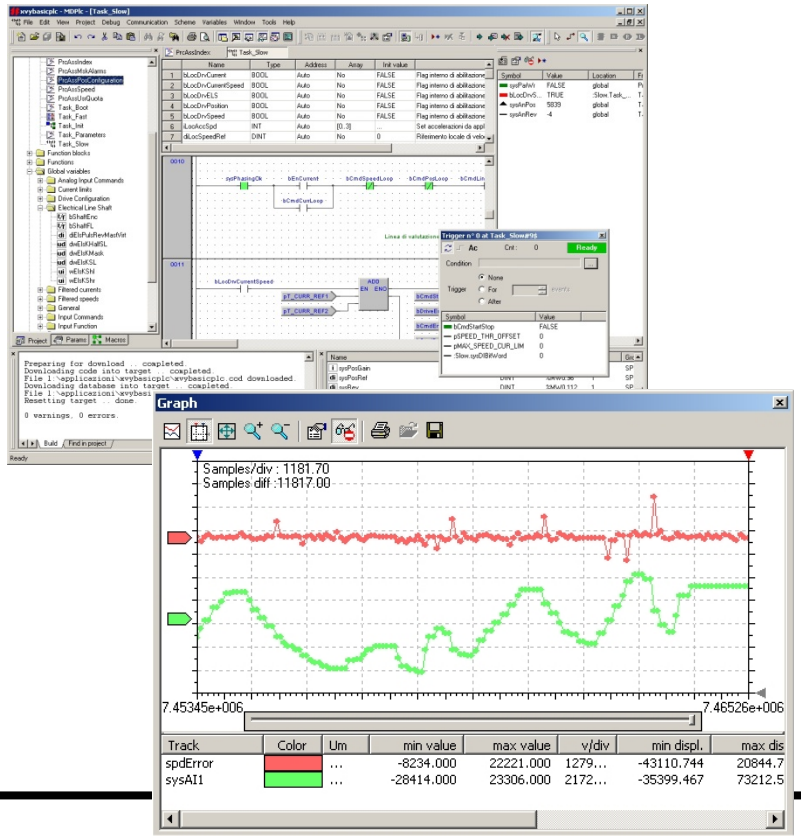
Modular servo drives



AXV300

English

How to write the applications with the MDPlc on AXV300

GEFRAN

Index of changes

Revision	Date	Author	Description
0.0	13-10-15	Grande P.	First edition, derived from AXV300 Mdplc V1_x_3_0_EN
0.1	16-1-17	Tonazzo P.	Changes on chapters 2 and 8 according to AXV300 Mdplc V3_x_0_0

Hardware / firmware configuration and manual compatibility

Module	HW	FW	Refer to manual	
			File name	Revision
AXV300 CU	1.0	0.0.0.16		
AXV300 CU	1.0	1.0.3.0	AXV300 Mdplc V1_x_3_0_EN	13-1-14
AXV300 CU	1.0	2.0.3.0	AXV300 Mdplc V2_x_3_0_EN	13-10-15
AXV300 CU	1.0	3.0.0.0	AXV300 Mdplc V3_x_0_0_EN	16-1-17

Thank you for choosing this Gefran product.
We will be glad to receive any possible information which could help us improving this manual. The e-mail address is the following: techdoc@ gefran.com.
Before using the product, read the safety instruction section carefully.
Keep the manual in a safe place and available to engineering and installation personnel during the product functioning period.
Gefran S.p.A has the right to modify products, data and dimensions without notice.
The data can only be used for the product description and they can not be understood as legally stated properties.

Contents

1 DEFINITION OF AN APPLICATION

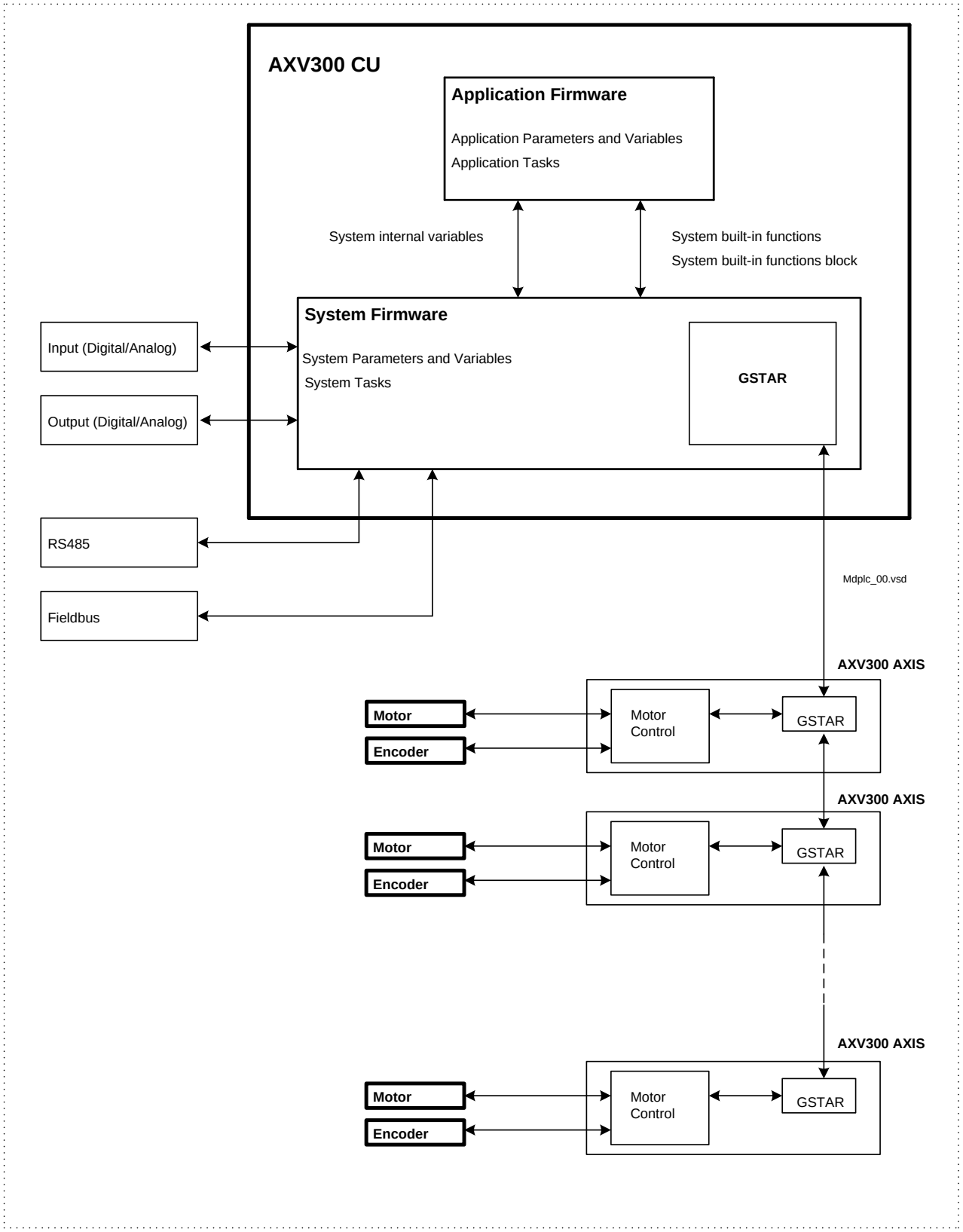
Using the MDPlc developing environment the user can write and download an application on the AXV300 system.
The application is defined by:

- **Tasks** where the user can write code using the 5 available IEC-61131 languages
- **Parameters and variables defined by the user** which can be organized in menus to be accessed by the tasks

Together with the parameters defined by the user, the drive has:

- Pre-existent **system parameters and variables** organized in hierarchical menus, which allow the drive basic configuration.
- Pre-existent **system internal variables** which allow to control and manage the drive different functions such as the control mode, the inputs, the outputs, the encoders, etc.

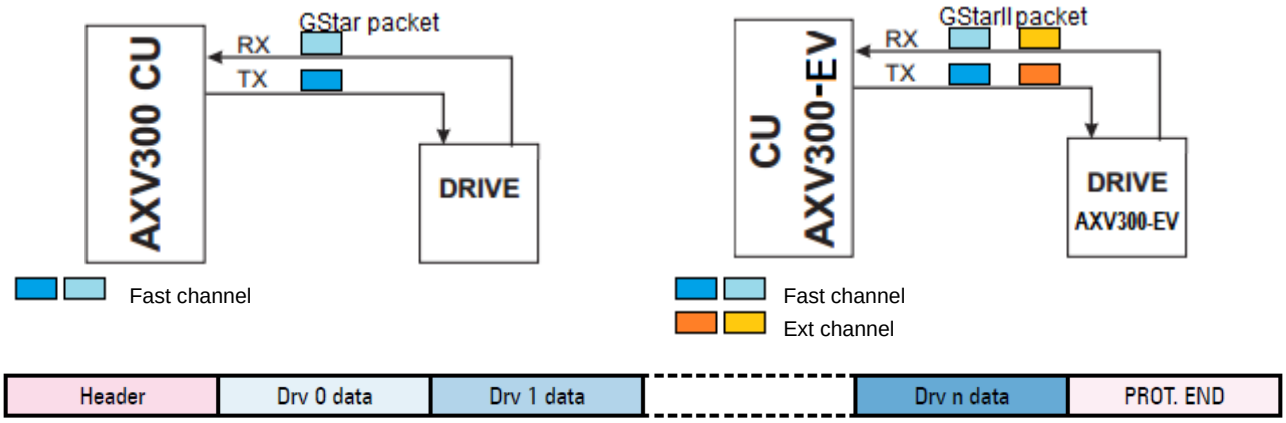
It is important to notify that axis are connected with AXV300 CU only by means GSTAR communication, so it is very important to know how it works and how to configure it to obtain user motion application.


AXV300 CU
Application Firmware
Application Parameters and Variables
Application Tasks
System internal variables
System built-in functions
System built-in functions block
System Firmware
System Parameters and Variables
System Tasks
GSTAR
Input (Digital/Analog)
Output (Digital/Analog)
RS485
Fieldbus
Mdplc_00.vsd
AXV300 AXIS
Motor
Encoder
Motor Control
GSTAR
AXV300 AXIS
Motor
Encoder
Motor Control
GSTAR
AXV300 AXIS
Motor
Encoder
Motor Control
GSTAR

2 GSTAR COMMUNICATION

Characteristics of the **GStar** communication system:

- High-speed serial communication with optical fibre loop (link)
- 2 communication links (GS1 and GS2)
- Supports up to 4 axis modules per link
- Basic cycle time of 250 us synchronised with the fastest system task
- Enables synchronisation of motor control tasks resident on axis modules
- 16-bit data transfer and fully configurable mappable data
- Different types of data exchange for real-time and service data
- Additional 7-word “GStarII” package exchanged at each step (only for system that use ONLY axis modules in the **AXV300-EV** series, **AXV300-CU** modules with FW version ≥ v3.0.0, GStar communication FPGA version ≥ V3.1.0, and all axis modules with FW version fw ≥v2.00):



Data package composition managed by a link in TX and RX (*)

(*) On all-AXV300-EV systems, a second “Ext” package of the same size is exchanged every 250us.

Control Word	Word 0 TX	Word 1 TX	Word 2 TX	Word 3 TX	Word 4 TX	Service channel reserved

Composition of “GStar” data sent at each step (250 us) for a single drive in the link

Control Word	Word 0 RX	Word 1 RX	Word 2 RX	Word 3 RX	Word 4 RX	Service channel reserved

Composition of “GStar” data received at each step (250 us) by a single drive in the link

Word 0 TX	Word 1 TX	Word 2 TX	Word 3 TX	Word 4 TX	Word 5 TX	Word 6 TX

Composition of additional “GStarII” data transmitted at each step (250us) for a single drive in the link

Word 0 RX	Word 1 RX	Word 2 RX	Word 3 RX	Word 4 RX	Word 5 RX	Word 6 RX

Composition of additional “GStarII” data received at each step (250us) by a single drive in the link

For more detailed information, see the “installation user manual”.

2.1 BASIC CONFIGURATION

To perform basic communication system configuration, enter the number of drives connected per optical fibre link in the parameters:

IPA	Name
1	Net 0 drives
2	Net 1 drives

The axis modules, starting from the last in the link, are associated with a number starting from 0 and increasing by 1 for each axis up to a maximum of 3 (see figure below).
The same applies for the second link, except that the numbers start from 4 instead of 0.
The drive's position in the **GStar** link does not affect operation.
This method unequivocally associates each axis module with the configuration parameters of the drives available on the control module.

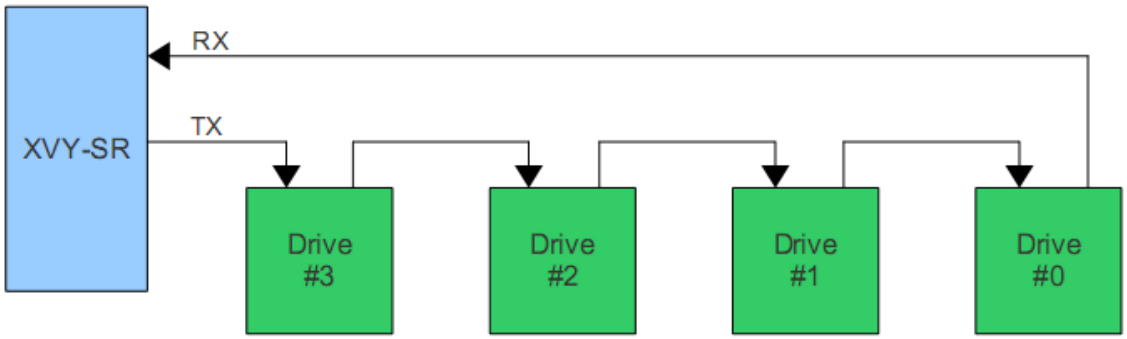


Figura 1 – Numerical association based on the drive's position in GStar link

2.2 GSTAR LINK CHANNELS

The GStar link is managed to provide 3 communication channels:

- Fast channel
- Ext channel (GStarII)
- Slow channel
- Service channel

N.B.: The Ext Channel is available only for an “all-AXV300-EV” configuration in which the CU has firmware version ≥v3.0.0 and all drives ≥v2.00. In addition, the CU must have FPGA version ≥v3.1.0. In all other cases, the additional “GStarII” package (and therefore the Ext Channel) are not enabled.

Each GStar package (not of Ext channel) also always includes a Control word (TX direction) and a Status word (RX direction).
Error management is based on CRC calculation for the entire link communication frame. An error in one package make wrong the entire link communication cycle.

2.2.1 Fast channel

This channel enables bi-directional and real-time transmission of up to 5 data words to all drives every 250 ms.
For all-AXV300-EV configuration: if extended mode is enabled in parameter X798 **DrvX ExtMode**, 6 data words are always available, regardless of parameter X801 **DrvX fast num objs**.

2.2.2 Ext channel

Available only for all-AXV300-EV configuration. Allows exchange of up to 7 additional data words every 250us, which the drive always processes in the following step compared to Fast words.
It has two work modes selected via parameter P.X798 **DrvX ExtMode** (N.B.: the parameter is on the "MAP EXT" menu, which is shown only for AXV300-EV drives. It is not useful for, and is not shown for, drives with previous firmware):

- Extended mode = Off
All 7 additional words are programmable and are exchanged every 250us, but in a following step compared to Fast words. The Fast and Slow channels act as in 2.3 "Configuration of real-time channels".

- Extended mode = On
The Slow channel has up to 26 words available and is inserted in Ext words instead of in Fast words. Therefore, the number of programmable Fast words depends on the number of Slow words used by the parameters on the "MAP DATA" menu:
X800 **DrvX slow per**
X801 **DrvX fast num objs**.

Therefore, in this mode the latter parameter refers to Ext words, not to Fast words.
The Service channel is also inserted along with the Ext words, but placed at the end of all Slow channel words.
The Fast channel always has 6 words available at 250us, regardless of X801 **DrvX fast num objs**.

2.2.3 Slow channel

This channel is based on the use of words not used by the fast channel. It enables bi-directional and real-time transmission of up to 16 data words to all the drives, with a cycle step that is a multiple of the fast channel.
For all-AXV300-EV configuration: if enabled, "Ext Mode" is On, up to a maximum of 26 data words with a cycle step that is a multiple of the Ext channel.

2.2.4 Service channel

This is a bi-directional non-real-time channel and does not depend on the configuration of the other channels. It is used by the control module to configure the drives and read their status in all operating modes.

2.3 Configuration of real-time channels

The fast and slow real-time channels make drive data available to the AXV300 CU module and send references to the axes. The variables to map in GStar may therefore differ because they depend on the motion application to be implemented and are specific to each drive.
Configuration is performed by setting, for each drive:

2.3.1 Extended mode = Off

A) the following two parameters:

X800 **Drvx slow per**
X801 **Drvx fast num objs**

IPA	Name	Description
X800	Drvx slow per	Slow channel cycle time (min 500 ms, max 4 ms)
X801	Drvx fast num objs	Number of words used in the fast channel (min 0, max 5)

Where x identifies the drive's position in the optical link, and X = (x +1).
These two parameters unequivocally define the word exchange mode to be used by GStar.

B) configuring the variables to exchange in parameters:

Fast RX:

IPA	Name	Description
X810	Drvx RX 0 obj	Drive x datum to exchange in fast mode in position 0
X811	Drvx RX 1 obj	Drive x datum to exchange in fast mode in position 1
X812	Drvx RX 2 obj	Drive x datum to exchange in fast mode in position 2
X813	Drvx RX 3 obj	Drive x datum to exchange in fast mode in position 3
X814	Drvx RX 4 obj	Drive x datum to exchange in fast mode in position 4

Fast TX:

IPA	Name	Description
X820	Drvx RX 0 obj	Drive x datum to exchange in fast mode in position 0
X821	Drvx RX 1 obj	Drive x datum to exchange in fast mode in position 1
X822	Drvx RX 2 obj	Drive x datum to exchange in fast mode in position 2
X823	Drvx RX 3 obj	Drive x datum to exchange in fast mode in position 3
X824	Drvx RX 4 obj	Drive x datum to exchange in fast mode in position 4

Slow RX:

IPA	Name	Description
X830	Drvx RX slow 0 obj	Drive x datum to exchange in Slow mode num 0
X831	Drvx RX slow 1 obj	Drive x datum to exchange in Slow mode num 1
X832	Drvx RX slow 2 obj	Drive x datum to exchange in Slow mode num 2
..	.. ..	
X849	Drvx RX slow 19 obj	Drive x datum to exchange in Slow mode num19

Slow TX:

IPA	Name	Description
X850	Drvx TX slow 0 obj	Drive x datum to exchange in Slow mode num 0
X851	Drvx TX slow 1 obj	Drive x datum to exchange in Slow mode num 1
X852	Drvx TX slow 2 obj	Drive x datum to exchange in Slow mode num 2
..	.. ..	
X869	Drvx TX slow 19 obj	Drive x datum to exchange in Slow mode num 19

Only for all-AXV300-EV systems (CU version fw ≥ v.3.0.0 and drive fw ≥ v2.00. For drives with previous version, the following parameters are not useful and are not shown):

IPA	Name		IPA	Name	Description
X930	Drv0 RX slow 20 obj”	…	X935	Drv0 RX slow 25 obj	not used
X950	Drv0 TX slow 20 obj”	…	X955	Drv0 TX slow 25 obj	not used

Ext TX

IPA	Name	Description
X870	Drvx RX ext 0 obj	Drive x datum to exchange in Ext mode num 0
…		
X876	Drvx RX ext 5 obj	Drive x datum to exchange in Ext mode num 5

Ext RX

IPA	Name	Description
X880	Drvx TX ext 0 obj	Drive x datum to exchange in Ext mode num 0
…		
X886	Drvx TX ext 5 obj	Drive x datum to exchange in Ext mode num 5

The configuration of these parameters is very important to ensure correct application functioning. For this reason this is not usually done manually by the operator, but implemented in the boot task of the user PLC program (see the MdPLC programming guide).

2.3.2 Example only-GStar configuration, applicable to AXV300, AXV300-EV, and mixed systems.

Allows on any system, even with a drive with fw < 2.00 or CU with fw < 3.0.0 or FGPA < v3.1.0).

IPA	Name	Description
1800	Drv0 slow per	1ms
1801	Drv0 fast num objs	2

This means the slow data cycle is repeated every 1 ms (4 250 ms steps) and the data exchanged are:
GStar TX packages:

Package No.	Data cycle	Word 0	Word 1	Word 2	Word 3	Word 4
1	1	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 0 obj	Drv0 TX Slow 1 obj	Drv0 TX Slow 2 obj
2	2	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 3 obj	Drv0 TX Slow 4 obj	Drv0 TX Slow 5 obj
3	3	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 6 obj	Drv0 TX Slow 7 obj	Drv0 TX Slow 8 obj
4	4	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 9 obj	Drv0 TX Slow 10 obj	Drv0 TX Slow 11 obj
5	1	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 0 obj	Drv0 TX Slow 1 obj	Drv0 TX Slow 2 obj
..	..	..	..	..		..

Drv 0 Control Word	Drv 0 TX 0 Obj	Drv 0 TX 1 Obj	Drv 0 TX 0 Slow 0 Obj	Drv 0 TX 0 Slow 1 Obj	Drv 0 TX 0 Slow 2 Obj	Service channel reserved
Drv 0 Control Word	Drv 0 TX 0 Obj	Drv 0 TX 1 Obj	Drv 0 TX 0 Slow 3 Obj	Drv 0 TX 0 Slow 4 Obj	Drv 0 TX 0 Slow 5 Obj	Service channel reserved
Drv 0 Control Word	Drv 0 TX 0 Obj	Drv 0 TX 1 Obj	Drv 0 TX 0 Slow 6 Obj	Drv 0 TX 0 Slow 7 Obj	Drv 0 TX 0 Slow 8 Obj	Service channel reserved
Drv 0 Control Word	Drv 0 TX 0 Obj	Drv 0 TX 1 Obj	Drv 0 TX 0 Slow 9 Obj	Drv 0 TX 0 Slow 10 Obj	Drv 0 TX 0 Slow 11 Obj	Service channel reserved

Composition of packages transmitted according to the configuration in the example

Data are exchanged in the RX direction in the same way.

Drv 0 Control Word	Drv 0 RX 0 Obj	Drv 0 RX 1 Obj	Drv 0 RX 0 Slow 0 Obj	Drv 0 RX 0 Slow 1 Obj	Drv 0 RX 0 Slow 2 Obj	Service channel reserved
Drv 0 Control Word	Drv 0 RX 0 Obj	Drv 0 RX 1 Obj	Drv 0 RX 0 Slow 3 Obj	Drv 0 RX 0 Slow 4 Obj	Drv 0 RX 0 Slow 5 Obj	Service channel reserved
Drv 0 Control Word	Drv 0 RX 0 Obj	Drv 0 RX 1 Obj	Drv 0 RX 0 Slow 6 Obj	Drv 0 RX 0 Slow 7 Obj	Drv 0 RX 0 Slow 8 Obj	Service channel reserved
Drv 0 Control Word	Drv 0 RX 0 Obj	Drv 0 RX 1 Obj	Drv 0 RX 0 Slow 9 Obj	Drv 0 RX 0 Slow 10 Obj	Drv 0 RX 0 Slow 11 Obj	Service channel reserved

Composition of packages received according to the configuration in the example

It is important to note that the slow and fast channels are closely connected.
In this configuration, a maximum of 12 words can be exchanged via the slow channel.
The general rule is as follows:
Max Slow Objs = (5 - Num Fast Obj) * (Slow Per / 250 us);

In the example
Max Slow Objs = (5 - 2) * (1 ms / 250 us) = 3 * 4 = 12.

2.3.3 Example of GStarII in extended mode = Off, only with all-AXV300-EV system

As in example 2.3.2, plus the additional 7+7 words of the Ext channel can be used. These words can be freely mapped or left unused.

Word 0 TX	Word 1 TX	Word 2 TX	Word 3 TX	Word 4 TX	Word 5 TX	Word 6 TX
Word 0 RX	Word 1 RX	Word 2 RX	Word 3 RX	Word 4 RX	Word 5 RX	Word 6 RX

Composition of data in additional Ext package

2.3.4 Example of GStarII in extended mode = On, only with all-AXV300-EV system

In this mode, the Fast channel always has 6 words available, regardless of how the Slow and Ext channels are programmed.

Control Word	Word 0 TX	Word 1 TX	Word 2 TX	Word 3 TX	Word 4 TX	Word 5 TX
Status Word	Word 0 RX	Word 1 RX	Word 2 RX	Word 3 RX	Word 4 RX	Word 5 RX

Composition of Fast package with Extended mode = "On"

The Slow channel is inserted in the Ext channel along with the Service channel.

Keeping the same settings of example 2.3.2:
1800 **Drv0 slow** for 1ms
1801 **Drv0 fast num objs** 2, now referred to Ext channel.
This means the Slow data cycle is repeated every 1ms (4 250us steps) and the exchanged data are:

GStarTX packages:

Package No.	Data cycle	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6
1	1	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 0 obj	Drv0 TX Slow 1 obj	Drv0 TX Slow 2 obj	Drv0 TX Slow 3obj	Drv0 TX Slow 4obj
2	2	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 5 obj	Drv0 TX Slow 6obj	Drv0 TX Slow 7 obj	Drv0 TX Slow 8 obj	Drv0 TX Slow 9obj
3	3	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 10obj	Drv0 TX Slow 11 obj	Drv0 TX Slow 12 obj	Drv0 TX Slow 13 obj	Drv0 TX Slow 14 obj
4	4	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 15obj	Drv0 TX Slow 16obj	Drv0 TX Slow 17obj	Drv0 TX Service 0	Drv0 TX Service 1
5	1	Drv0 TX 0 obj	Drv0 TX 1 obj	Drv0 TX Slow 0 obj	Drv0 TX Slow 1 obj	Drv0 TX Slow 2 obj	Drv0 TX Slow 3obj	Drv0 TX Slow 4obj
..	..	..	..	..	..	..	..	..

Data are exchanged in the RX direction in the same way.
In this case, the limit is:
Max Slow Objs = (7- **Num Fast Obj**) * (Slow Per / 250 us) – **Num word service;** where there are always 2 service words.

In the example
Max Slow Objs = (7- 2) * (1 ms / 250 us) -2= 5 * 4 -2= 18.

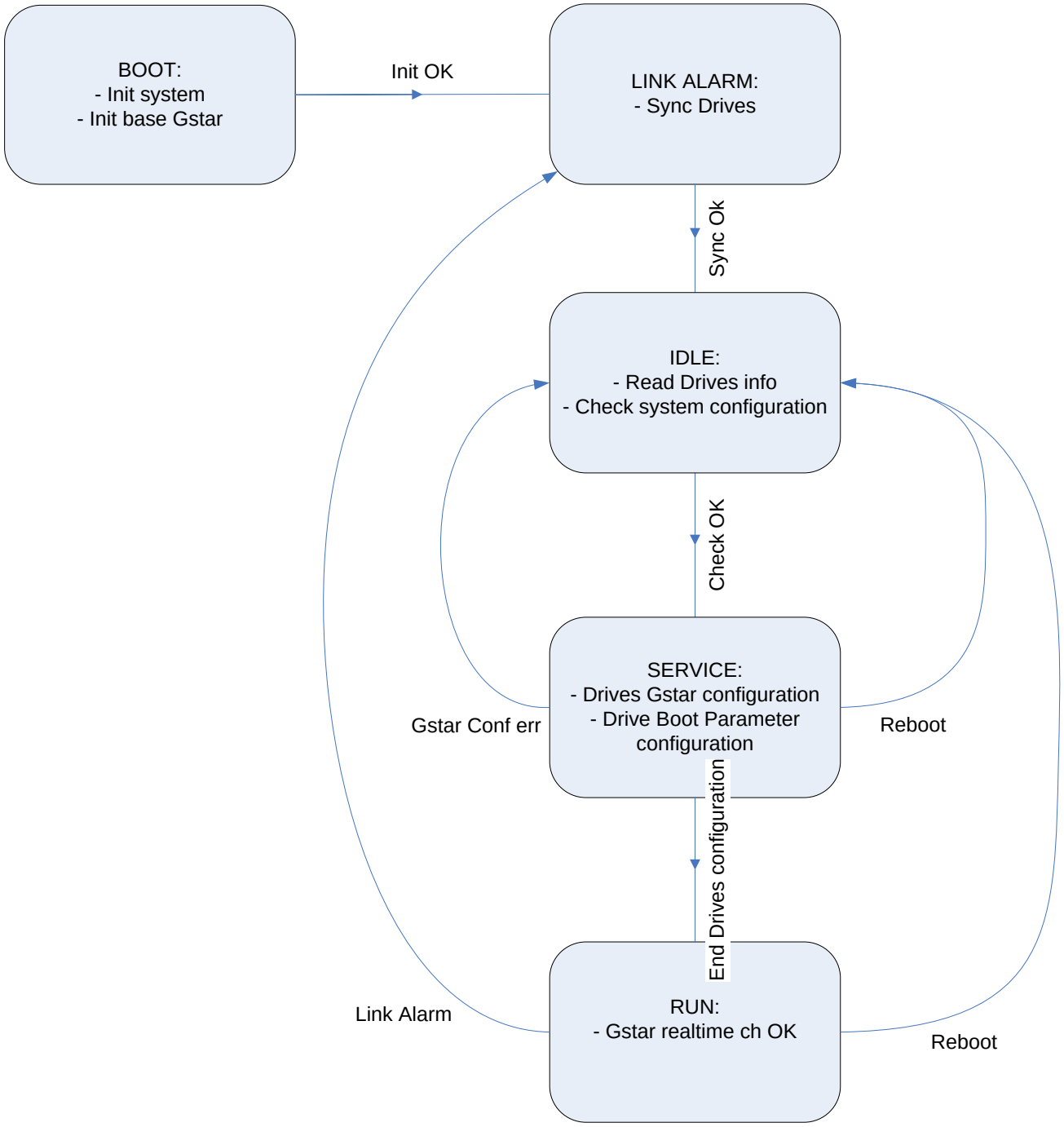
The Maximum number of slow words that can be inserted is sized by **Num Fast Obj** = 0 at 1ms
Max Slow Objs = (7-0) * (1 ms / 250 us) -2 = 7 * 4 -2 = 26

2.4 32 bit variables

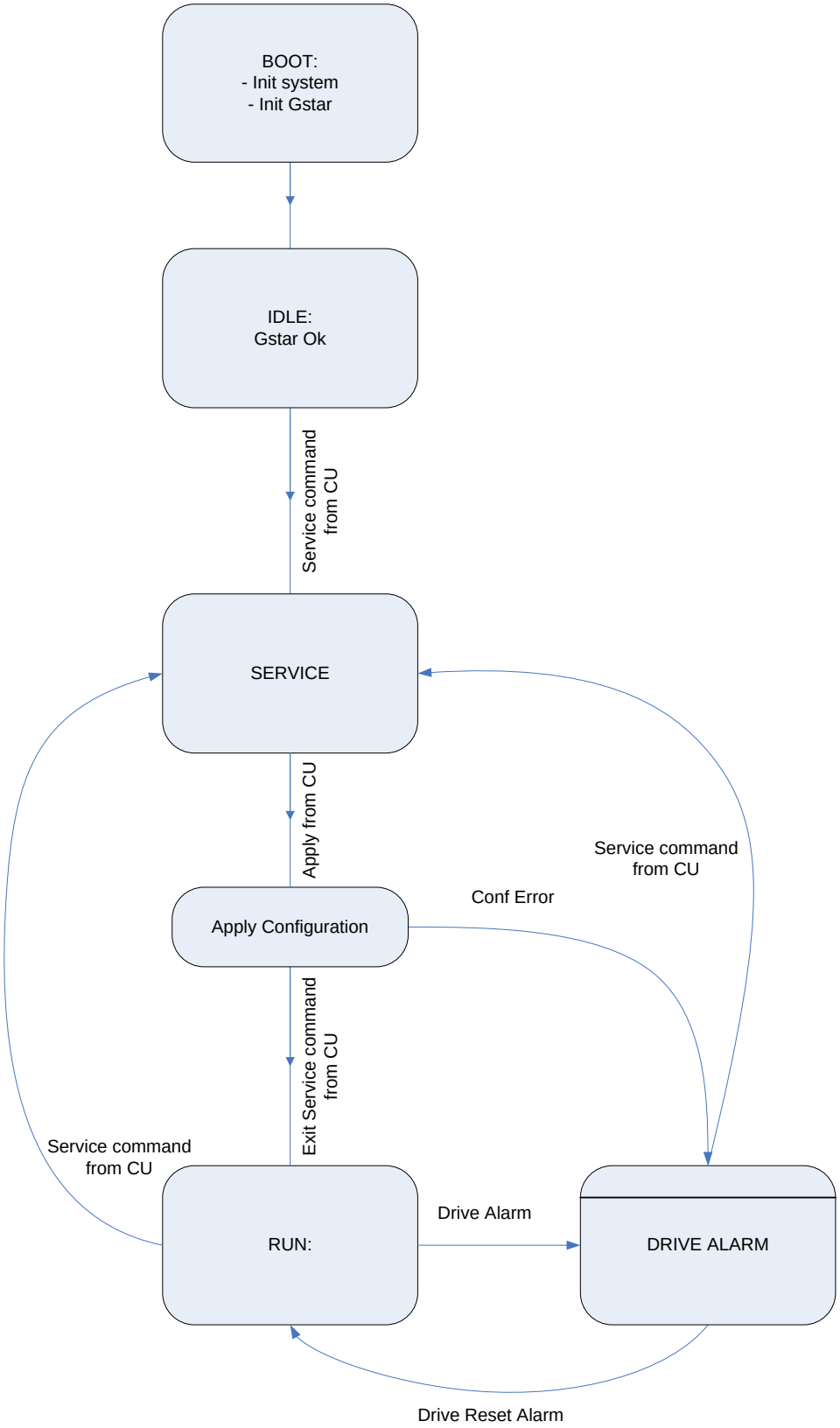
32-bit variables are mapped by breaking them down into two 16-bit parts, with the low part of the 32-bit variable ending in Lo and the high part in Hi.
To ensure time consistency of the 32-bit variables (e.g.: speed) these two components must be mapped in the same **GStar** package.
Particular attention must therefore be paid when mapping 32-bit variables on the slow channel.
If the Ext channel is used, the two components cannot be in different channels. Therefore, for example, one in the Fast channel and the other in the Ext channel, or one in the Slow channel and the other in the Ext channel.

3 SYSTEM STATUS

3.1 AXV300 CU



3.2 AXIS MODULE



3.3 DRIVE CONTROL LOOP CONFIGURATION

3.3.1 AXV300 drive motor control Scheme

4 TASKS

In the task the user writes source code in order to perform the required functions. Inside the task it is possible to enter all user parameters and all system internal variables.

The tasks available on the AXV300 system are:

Name	Type and execution time	Description
Boot	Asynchronous. It is performed only once immediately after the start up before system initialization and any other task	It is used for initialization operations, to set Gstar communication parameter, etc.
Init	Asynchronous. It is performed only after axv300 system initialization and just before cyclic task.	It is used for initialization operations, etc.
Slow	Synchronous. 1-8 msec to be set through the **Slow Task Period** parameter (IPA n° 3).	It is used for real time checks and for the control logic
Fast	Synchronous at 250 us.	It is the fastest task and it is used for real time checks.
Background	Asynchronous	It is used for slow and asynchronous operations.
Parameter	Asynchronous. It is performed every time the user changes a drive parameter	It is useful to perform specific operations after the modification of a system parameter

4.1 TASKS EXECUTION

At the drive start up or reset the **Boot and Init** task is enabled allowing to perform the first initializations.
Task **Slow, Fast** and **Background** are performed too.
At AXV300 system start up, or after a reset, the **Boot** task is performed, where it is possible to perform the first initialization operations.
After Boot task firmware uses GStar settings to initialize GStar communication.
After this firmware launches **Init** task that can be used to read Init status of the system.
After firmware launches cyclic tasks **Fast** and **Slow** that can use all firmware data structures.

The code of AXV300 CU is divided into three tasks which are listed in the following table according to their priority level:

Task	Execution period	Max Execution time	Priority (0 = max)
Fast	250 μs	< 190 μs	0
Slow	1 – 8 ms	< 90%	1
Background	Asynchronous	--	2

Table 1: Tasks performed by the CU (according to their priority level)

There is also the Parameter task which is performed every time a parameter is modified.

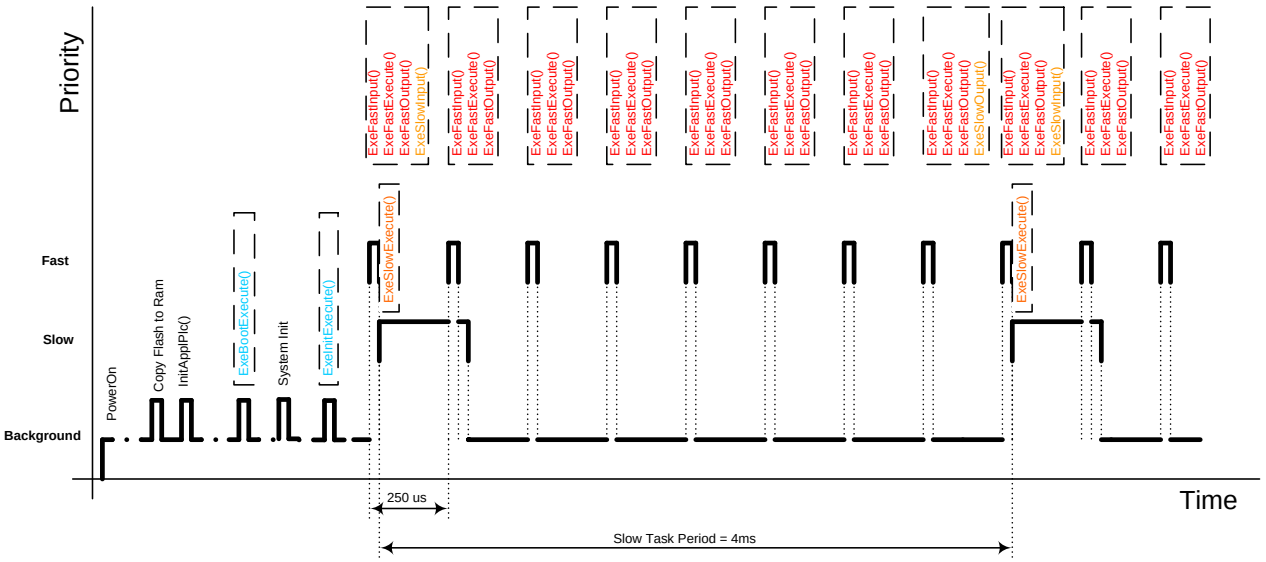


Figure 2 - AXV300 CU system tasks execution (example with Slow Task Period = 4ms)

1 Fast Task

The firmware executes Fast task inside other firmware functions as shown below:

- ReadGSTARdata();
- ReadLocalIO();
- FieldBusInput();
- TaskFast();
- TaskSlowManagement: SlowTaskInput() or SlowTask Output() or start SlowTaskExecute interrupt;
- WriteGSTARdata();
- WriteLocalIO();
- FieldBusOutput();
- TaskDurationVerify();

Duration of these functions cannot be over 180 µs.

2 Slow Task

The task is divided into 3 phases:

Phase	Max. execution time	Execution period
Slow IN inside Fast task	250 us	1-8 ms
Slow EXE executed by means an interrupt request	90% of 1-8 ms	1-8 ms
Slow OUT inside Fast task	250 us	1-8 ms

Duration of SlowTask cannot be over 90% of SlowTaskPeriod (Ex: Slow Task Period = 4ms, Max SlowTask duration = 3.6 ms).

5 Parameters and Variables defined by the User

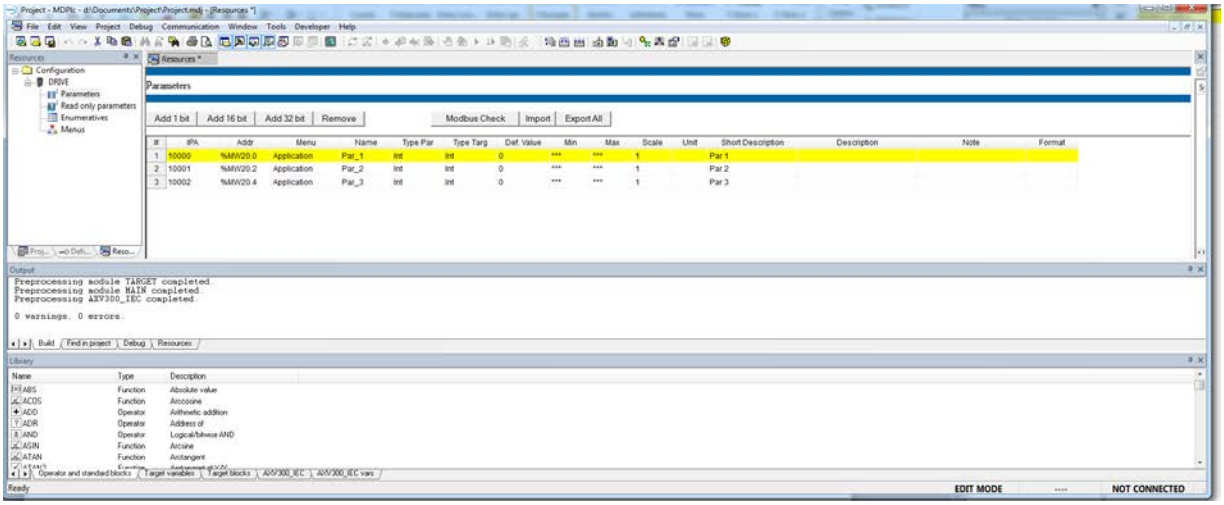
The user defined parameters and variables (these are also defined as read only parameters) are specific for the application. They all have a parameter index (IPA) grather than 10000. They can be organized in any menu.
The AXV300 firmware supplies 2 data blocks for the parameter declaration.
Each data block contains parameters with different formats and features.
Each data block is combined to a range of parameter indexes.
The management, the assignment of the parameter indexes and the specific menu are automatically managed by MDPlc.
The parameter, contrary to the variable, can be modified and can be saved permanently in the drive flash.
Here are the parameter and variables properties:

Data Block	Parameter type	Parameters number	Parameter index (IPA)	It can be modified and saved in FLASH
20	Any	2048	10000	YES

Data Block	Variable type	Parameters number	Parameter index (IPA)	It can be modified and saved in FLASH
21	Any	1024	14000	NO

Inside the tasks it is possible to enter the user parameters and variables placing the prefix p before the parameter name or the prefix v before the variable name.

Example:



As shown in figure Par_1 can be used into task using pPar_1 label.
The definition of the parameter has be done using Parameter item into tab Resource.
The fields of the record are shown into table below.

Record field	Description	Note
IPA	Identifier parameter address	It is autocalculated on click “Add xx” html buttons
Addr	Memory address	It is autocalculated on click “Add xx” html buttons
Menu	menu	It can be select one menu defined into Menus item of Resource Tab on the left
Name	Internal name of parameter	This is used into task by means pName label

Type Par	External type of parameter	This is used by Gfexpress configurator and Keypad
Type Tar	Target type of parameter	Database internal type
Def. Value	Default value	Used on “Load default parameter command”
Min	Minimum value	
Max	Maximum value	
Scale	Scaling factor	It is used to scale external value and internal value
Unit	Measure unit of parameter	It is used on keypad and configurator
Short Description	Short description of name parameter (*)	It is used on keypad and configurator
Description	description	It is used on configurator
Note	note	It is used on configurator
format	Number format	It is used on Keypad and configurator: %.4f = 0.0000 %.3f = 0.000 %.2f = 0.00 %.1f = 0.0 %.0f = 0 %f = automatic %08x = 00000000H %04x = 0000H %02x = 00H %x = automatic

(*) Because these strings are shown on keypad, there are some rules on max length: - menu

* max 18 char at entrypoint menu
* max 15 char at sublevel 1
* max 12 char at sublevel 2
* max 12 char at sublevel 3

- Short description
  * max 20 char
- Unit
  * max 5 char
- Enumerative
  * max 14 char for the label

If length is bigger than limit then keypad shows only the first chars up to the limit.

In same way for user read only parameters (variables). The variables can be used into task using vName label.
The means of the record fields are the same of parameters.

6 Menu defined by the user

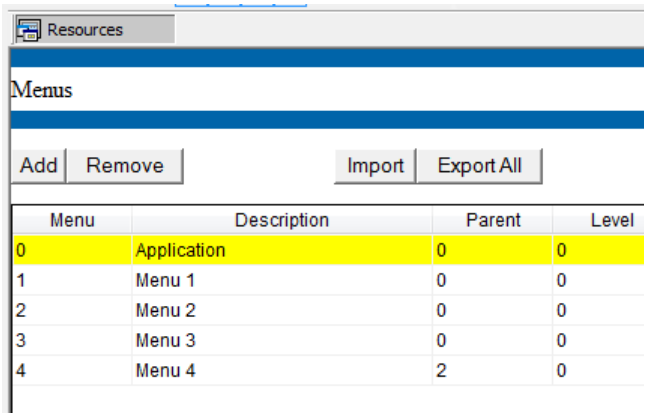
Into Mdplc, the user can create user menu. There are following limitations about how many children menu are possible and their composition:

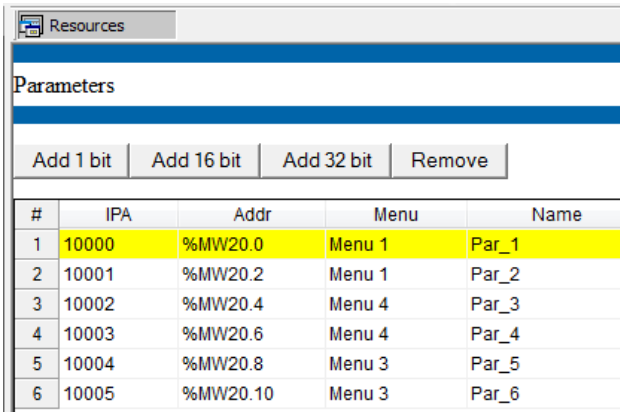
- a menu can contain up to 255 of only parameters and variables or up to 255 of only menus submenus, it is not possible to have menu and parameter into same menu
- it is possible to set up to 2 level of submenu.

Example of a possible parameter menu tree of an User Application

-Application
+Menu1
| +Par 1
| +Par 2
| +..
+Menu2
| +Menu4
| +Par 3
| +Par 4
| +..
+Menu3
| +Par 5
| +Par 6
| +..
+..

The definitions of menus and parameters to obtain the tree of parameters of the example are the following.





7 System Parameters and Variables

The system parameters are parameters available in every application and are part of the AXV300 firmware. They are organized according to hierarchical menus and they all depends on the "Main Menu" menu. They configure the drive basic functioning procedure.
Please refer to the AXV300 User Manual to have info about system parameters.
In this manual will be given only the parameters relevant to programming the system AXV300. Refer to the User installation manual for the description of all parameters.

7.1 MAIN CONFIG

PARAMETER	No.	Format	Unit
Net 0 Drives	1	UnsignedShort	--
Net 1 Drives	2	UnsignedShort	--
Slow Task Period	3	Enum	--

7.2 MONITOR\SYSTEM

PARAMETER	No.	Format	Unit
Fast task time	710	Float	ms
Slow task time	712	Float	ms

7.2.1 Fast task time [ms]

This parameter measures the period of time needed to perform the fast task. The Fast task is performed every 250us.

7.2.2 Slowt task time [ms]

This parameter measures the period of time needed to perform the slow task. The slow task is performed according to the specifications of the Slow Task Period parameter.

8 System Internal Variables

Using the MDPlc compiler supplied by GEFRAN it is possible to have access to several system variables of the AXV300 multidrive system.
These variables are listed in the '**AXV300_IEC vars'** windows inside the MDPlc development environment (Library toolbar).

Many variables are available as structures of data:
- 8 drive data structures that are used to manage and control all the possible configurated drives on GSTAR nets.
- GSTAR status data structure
- system status data stucture

It is important to notice that the compiler can create a process image of the variables. In the AXV300 target variables are usually without process image.
The following paragraphs describe the different variable logic groups according to specific subjects.

8.1 GSTAR DATA

Name	Type	Description	Access
sysGStar	STRUCT GStarControl	Drives Control Word struct	Read/Write

Name	Type	Description	Access
linkOk	BOOL	Basic link functions ok: hardware flag	Read Only
linkAlarm	BOOL	Link not Ok: hardware and software flag	Read Only
bootRequired	BOOL	Reboot Command	Read/Write
dataExchange	BOOL	Link Ok and data exchange with drive active	Read Only
offlineMode	BOOL	If TRUE drive parameter are not exchanged with drive (expert only)	Read/Write
bDrivesCfgTake	BOOL	If set TRUE then firmware takes system net configuration	Read/Write
bDrivesCfgOk	BOOL	It is TRUE if there is no problem on check net configuration	Read Only
bDriveCfgOk	ARRAY [0..7] OF BOOL	This array report the result of the check configuration: a bit for each axis	Read Only
bLinkInit	BOOL		Read Only
bLoadDefault	BOOL	If set TRUE then the firmware does an Load default parameter command	Read/Write
fpgaRxStatus	ARRAY [0..1] OF WORD	RESERVED	Read Only
numDrivesRx	ARRAY [0..1] OF UINT	Number of drives received from GSTAR data link on net 0 and net 1	Read Only
numDrivesCnf	ARRAY [0..1] OF UINT	Number of drives configured on GSTAR data link on net 0 and net 1	Read/Write
srvErrorOp	UINT	Monitor of operation of service channel: 0 – Read 1 – Write	Read Only
srvErrorCode	UINT	Error code of service channel: 0 – No error 1 – Link error 2 – Invalid Call 3 – Drive error 4 – Time Out	Read Only
srvErrorDrive	UINT	Drive number of the service channel that has given error	Read Only
srvErrorOID	UINT	Object exchanged that has given error	Read Only

Name	Type	Description	Access
GStarII Status	UINT	Status of GStarII Channel 0 – Off 1 – On See parameter 704 “GStarII Status”	Read Only
bExtMode	ARRAY [0..7] OF BOOL	TRUE if parameter X798 “DrvX Ext Mode” is “On”	Read Only

Additionally , the following words are available, to control functions Brake,Cool and Freeze capture with AXV300-EV axes.

Name	Type	Description	Access
sysDriveCW2	ARRAY OF STRUCT	Drive Control Word 2 struct	Write
sysDriveSW2	ARRAY OF STRUCT	Drive Status Word 2 struct	Read Only

Important note:
sysDriveCW2 and sysDriveSW2 are not exchanged by default : it is necessary to map them as one of the Fast , Slow , or Ext words to use the Brake,Cool or Freeze capture functions.

At Startup link Ok goes TRUE after drives synchronization, dataExchange goes TRUE when drives are configured and process data exchange between CU and Axis is active.
So at startup it needs verify dataExchange flag to begin drives control.

When dataExchange is active, it can also check if GStarII additional packets are exchanged with flag GStarIIStatus: if any parameter is configured in menu “MAP EXT”, GStarII may still not be active if a drive with firmware < v2.00 is in the link, and therefore some of the mapped variables may not be available.

8.2 SYSTEM STATUS

Name	Type	Description	Access
sysStatus	STRUCT	System status structure	Read Only

Name	Type	Description	Access
wStatus	UINT	System status	Read Only
dwAlarmCode	DWORD	System Alarm bitword	Read Only
dwBuildNumber	DWORD	Build number of firmware	Read Only
wSlowTaskPeriod	UINT	Slow task period in us	Read Only
dwFastTaskDuration	UDINT	Fast task duration in us	Read Only
dwSlowTaskDuration	UDINT	Slow task duration in us	Read Only
dwPlcCode	DWORD	Plc Alarm code	Read Only

8.3 DATABASE

Name	Type	Description	Access
sysDBase	STRUCT	Database structure	Read Only

Name	Type	Description	Access
bWr	BOOL	True = write operation, False = Read	Read Only
bNtfOnly	BOOL	True = only notify for system parameter	Read Only
bPreLoadDef	BOOL	Firmware are in “preload default” operation on database	Read Only
bLoadDef	BOOL	Firmware are in “load default” operation on database	Read Only
Ipamodify	UINT	IPA of parameter modifyed	Read Only
NtfEsi	DINT	Return value of notification procedure	Read Only
parFloat	REAL	value of parameter in type “float”	Read Only

Name	Type	Description	Access
parLong	DINT	value of parameter in type “long”	Read Only
parAttribute	DWORD	Attribute of the record of database	Read Only
parMin	REAL	Minimum value	Read Only
parMax	REAL	Maximum value	Read Only
parScale	REAL	Scale factor	Read Only
parOffs	REAL	Offset Value	Read Only

8.4 DRIVE DATA STRUCTURES

All the drive of AXV300 multi drive system are connected with CU module by means serial communication GSTAR. This allows data exchange at 250us cycle rate.
There are 8 drive data structures are available to manage and control drive functions.

These structures are listed below.

Name	Type	Description	Access
sysDriveCW	ARRAY OF STRUCT	Drives Control Word struct	Read/Write
sysDriveSW	ARRAY OF STRUCT	Drives Status Word struct	Read Only
sysDriveControl	ARRAY OF STRUCT	Drive Control Variables struct	Read/Write
sysDriveStatus	ARRAY OF STRUCT	Drive Status Variables struct	Read Only

8.4.1 SysDriveCW[x]

Name	Type	Description	Access
cmdEnable	BOOL	Drive x command Enable If set to TRUE Drive x, if not in alarm or service, immediately start PWM modulation and motor control. **Note!** If there is an AXV300 system alarm then cmdEnable is set by firmware at FALSE.	Read/Write
cmdAlarmReset	BOOL	Drives x command Alarm Reset If set to TRUE Drive x, if in alarm, acts a reset of alarms. **Note!** Drive read this command in trigger mode. This command has no effect if drive is in Configuration Alarm or drive is not in Alarm.	Read/Write
cmdEncIdxCapEn	BOOL	Drive x command to enable encoder capture.	Read/Write

Example:
ST PLC CODE for Drive 0

```
sysDriveCW[0].cmdEnable := TRUE;
```

8.4.2 SysDriveCW2

Name	Type	Description	Access
cmdBrakeOn	BOOL	Drive x Brake On command	Write
cmdEncFrzCapEn	BOOL	Drive x command to enable freeze capture	Write
cmdSpare1On	BOOL	Drive x Spare1 digital output	Write

8.4.3 SysDriveSW[x]

Name	Type	Description	Access
Enable	BOOL	Drive x enable status Refers if Pwm modulation, and motor control, is active.	Read Only
Alarm	BOOL	Drive x alarm status Refers if Drive x is in alarm. **Note!** wAlarm word of Drive Status structure shows alarm code.	Read Only
Ready	BOOL	Drive x ready status Refer if Drive x is present and initialized. **Note!** Ready do not depend on Drive alarm status.	Read Only
EncIdxOk	BOOL	Drive x encoder capture status. If true then drive store encoder position captured into sysDriveStatus data structure.	Read Only
TrqLim	BOOL	Drive is in Torque current limit	Read Only
SpdZero	BOOL	Motor is at zero speed Note! SpdZero is active if filtered speed <10 rpm	Read Only

8.4.4 SysDriveSW2

Name	Type	Description	Access
BrakeSts	BOOL	Drive x Brake Status	Read Only
EncFrzOk	BOOL	Drive x Freeze status if true then drive store freeze position in sysDriveStatus data structure	Read Only
CoolSts	BOOL	Drive x Motor cool status	Read Only
Spare2In	BOOL	Drive x Spare2 digital input status	Read Only

8.4.5 SysDriveControl[x]

All records are Read/Write type.

Name	Type	Description	Drive Object Index	Firmware scale
wFwSel	UINT	Drive x firmware selectors bitword Bit 0 – Reserved Bit 1 – ExtIntElTheta Bit 2 – ExtIntIdRef Bit 3 – ExtIntIqRef Bit 4 – ExtIntVdqRef Bit 5 - 15 – reserved	19	--
wSpeedRefLo	UINT	Drive x speed reference (Lo word)	14	(**)
wSpeedRefHi	UINT	Drive x speed reference (Hi word)	15	(**)
iIdRef	INT	Drive x flux current reference	903	Cnts2Arms
iIqRef	INT	Drive x Torque current reference	904	Cnts2Arms
iElThetaRef	INT	Drive x electrical theta reference	902	2^16 = 360°
iPosIqLim	INT	Drive x user maximum positive torque current reference	1302	Cnts2Arms
iNegIqLim	INT	Drive x user maximum negative torque current reference	1309	Cnts2Arms
iVdRef	INT	Drive x user direct voltage reference	905	Cnt2Vout
iVqRef	INT	Drive x user quadrature voltage reference	906	Cnt2Vout
wControlWord2	UINT	Drive x Control Word 2	3000	-

Examples

Drive 0 speed set point.
Variables

```
SpdRef = user parameter (type “REAL”, unit “rpm”)
Drv0SpeedRef = local variable (type “DINT” );
```

ST PLC Source Code

```
Drv0SpeedRef := pSpdRef * sysDriveStatus[ 0 ].rpm2cnts;
sysDriveControl[ 0 ].wSpeedLo := TO_UINT( Drv0SpeedRef AND 16#FFFF );
sysDriveControl[ 0 ].wSpeedHi := TO_UINT( SHR ( Drv0SpeedRef,16) );
```

Firmware Scale
Drv0SpeedRef : 2^31 = 120000 rpm

Example:
Drive 0 Torque current set point.

Variables

```
CurRef = user parameter (type “REAL”, unit “Arms”)
Drv0TrqRef = local variable (type “INT” );
```

ST PLC Source Code

```
Drv0TrqRef := TO_INT( pCurRef * sysDriveStatus[ 0 ].Arms2cnts);
sysDriveControl[ 0 ].iIqRef := Drv0TrqRef ;
```

Firmware Scale
Current scale depends on drive size, so use available conversion factor Arms2Cnts.
Approximately max drive overload current has value of 9680 cnts.

8.4.6 SysDriveStatus[x]

All records are READ ONLY.

Name	Type	Description	Drive Object Index	Firmware scale
wStatus	UINT	Drive x status 0 – Not configured 1 – IDLE 2 – SERVICE 3 – RUN 4 – ALARM 5 - RESERVED 5 – RESERVED 6 – UNKNOW	--	--
wFwRelease	UINT	Drive x firmware version (read only at startup) Ver = wFwRelease / 1000 Rel = wFwRelease / 10	--	--
wHwRelease	UINT	Drive x hardware version (read only at startup) Ver = wHwRelease / 1000 Rel = wHwRelease / 10	--	--
dwSizeCode	UDINT	Drive size code (read only at startup) Ex: 10413 – AXV300 10413 21020 – AXV300 21020	--	--

Name	Type	Description	Drive Object Index	Firmware scale
dwSerialNumberNumPart	UINT	Drive x Serial number, only numeric part (read only at startup)	--	--
dwSerialNumberCodePat	UINT	Drive x Serial Number, only code part (read only at startup)	--	--
dwFwName	UDINT	Drive x firmware name code (read only at startup)	--	--
fInom	REAL	Drive x nominal current (initialized by firmware after GSTAR drive recognizing)	--	--
fIovl	REAL	Drive x max current in overload (initialized by firmware after GSTAR drive recognizing)	--	--
fInom0Hz	REAL	Drive x nominal current at fout = 0 Hz (initialized by firmware after GSTAR drive recognizing)	--	--
iFswMin	INT	Drive x minimum switching frequency (initialized by firmware after GSTAR drive recognizing)	--	--
iFswMax	INT	Drive x maximum switching frequency (initialized by firmware after GSTAR drive recognizing)	--	--
Arms2cnts	REAL	Drive x conversion factor for current	--	--
cnts2rpm	REAL	Drive x conversion factor for motor speed	--	--
rpm2cnts	REAL	Drive x conversion factor for motor speed	--	--
cnts2Arms	REAL	Drive x conversion factor for current	--	--
cnts2Vdc	REAL	Drive x conversion factor for DC voltage	--	--
cnts2Vout	REAL	Drive x conversion factor for Output voltage	--	--
iEncARev	UINT	Drive x encoder revolutions	11	--
wEncAPosLo	UINT	Drive x encoder Position: lo word, incremental tracks	12	(1)
wEncAPosHi	UINT	Drive x encoder Position: hi word, incremental tracks	13	(1)
iEncAAbsRev	INT	Drive x absolute encoder revolutions	53	--
wEncAAbsPosLo	UINT	Drive x encoder Position: lo word, absolute tracks	2	(1)
wEncAAbsPosHi	UINT	Drive x encoder Position: hi word, absolute tracks	3	(1)
iEncAIdxRev	INT	Drive x encoder index revolution	50	--
wEncAIdxPosLo	UINT	Drive x encoder index position lo word	51	(1)
wEncAIdxPosHi	UINT	Drive x encoder index position hi word	52	(1)
iEncAIdxCapRev	INT	Drive x captured encoder index revolution	60	--
wEncAIdxCapPosLo	UINT	Drive x captured encoder index position lo word	61	(1)
wEncAIdxCapPosHi	UINT	Drive x captured encoder index position hi word	62	(1)
iActPosIqLim	INT	Drive x actual positive torque current limit	200	(3)
iActNegIqLim	INT	Drive x actual negative torque current limit	201	
iActIqRef	INT	Drive x actual torque current reference	17	(3)
iActIdRef	INT	Drive x actual flux current reference	18	(3)
wSpeedLo	UINT	Drive x motor speed: lower word	20	(2)
wSpeedHi	UINT	Drive x motor speed: higher word	21	(2)
wSpeedRefLo	UINT	Drive x motor speed reference: lower word	22	(2)
wSpeedRefHi	UINT	Drive x motor speed reference: higher word	23	(2)
wSpeedFiltLo	UINT	Drive x filtered motor speed: lower word	24	(2)
wSpeedFiltHi	UINT	Drive x filtered motor speed: higher word	25	(2)
iVdcLink	INT	Drive x DC voltage	103	1200 V = 2^15
iDrvTemp	INT	Drive x power module temperature	104	--
iFout	INT	Drive x output frequency	39	1000 Hz = 2^15
iDrvOvld	INT	Drive x actual overload percentage	40	100% = 2^14
iId	INT	Drive x act flux current	33	(3)

Name	Type	Description	Drive Object Index	Firmware scale
iIq	INT	Drive x act torque current	34	(3)
iIout	INT	Drive x output current	35	(3)
iIqFilt	INT	Drive x actual filtered torque current	43	(3)
iVout	INT	Drive x output voltage	36	(4)
wDriveAlarms	UINT	Drive x alarm bitword Bit 0 – Undervoltage Bit 1 – Overvoltage Bit 2 – Encoder alarm Bit 3 – Overtemperature/Undertemperature Bit 4 – PowerFail Bit 5 – Link Error Bit 6 – Motor Overtemperature Bit 7 – Motor Overspeed Bit 8 – Not used Bit 9 – Not used Bit .. – Not used Bit 12 – Fast overload of the drive Bit 13 – Slow overlaod of the drive Bti 14 – Not used Bit 15 – drives configuration error These Alarms can be reset by means Control Word bit command: cmdResetAlarm	600	--
wDriveWarnings	UINT	Drive x warnings bitword Bit 0 – Undervoltage Bit 1 – Overvoltage Bit 2 – Encoder alarm Bit 3 – Overtemperature/Undertemperature Bit 4 – PowerFail Bit 5 – Link Error Bit 6 – Motor Overtemperature Bit 7 – Motor Overspeed Bit 8 – Not used Bit 9 – Not used Bit .. Bit 12 – Fast overload of the drive Bit 13 – Slow overload of the drive Bit 14 - Encoder warning (this warning don't became alarm) Bit 15 – drives configuration error	602	--
wDriveConfError	UINT	Drive x configuration error: Bit 0 – PWM modulator error Bit 1 – encoder error Bit 2 – encoder Drive overload Bit 3..7 – Not used	610	--
iIu	INT	Drive Phase U current	30	(3)
iIv	INT	Drive Phase V current	31	(3)
iMotRev	INT	Drive x motor position	57	(5)
wMotPosLo	UINT	Drive x encoder Position: lo word, incremental tracks	58	(5)
wMotPosHi	UINT	Drive x encoder Position: hi word, incremental tracks	59	(5)
wMotTemp	UINT	Drive x motor temperature sensor resistance	105	1 = 1 Ohm
iActVoutLimit	INT	Drive x max output voltage limit	44	(4)
iEncAFrzRev	INT	Drive x encoder freeze revolutions	70*	--
wEncAFrzPosLo	UINT	Drive x encoder freeze position Lo word	71*	--
wEncAFrzPosHi	UINT	Drive x encoder freeze position Hi word	72*	--

Name	Type	Description	Drive Object Index	Firmware scale
iEncAFrzCapRev	INT	Drive x captured encoder freeze revolutions	80*	
wEncAFrzCapPosLo	UINT	Drive x captured encoder freeze position Lo word	81*	
wEncAFrzCapPosHi	UINT	Drive x captured encoder freeze position Hi word	82*	
wStatusWord2	UINT	Drive x Status Word 2	3100	-
wBrakeStatus	UINT	Drive x Brake status	1608 **	-
wMotCoolStatus	UINT	Drive x Motor cool status	1618 ***	-

* Encoder freeze function (available only for encoders with incremental tracks):
When there is an edge in the "Freeze" signal , according to the selection in X209 **DrvX Frz Edge Conf**, the position is immediately copied in iEncAFrzRev , wEncAFrzPosLo , wEncAFrzPosHi (parameters X768 **DrvX EncFrz Rev** and X766 **DrvX EncFrz Pos**).

** Bit EncFrzOk in sysSW2 is set to TRUE and the position is captured in variables iEncAFrzCapRev, wEncAFrzCapPosLo and wEncAFrzCapPosLo only if bit cmdEncFrzCapEn in sysDriveCW2 was set to TRUE.

*** Then to reset bit EncFrzOk, cmdEncFrzCapEn must be set to FALSE.

(1) POSITION SCALE
Encoder position into drive is normalized at 2^32, so a revolution is 2^32 counts:
wEncAposLo is lower word of 32 bit word data;
wEncAPosHi is higher word of 32 bit word data.

Example:

Variable
```
Pos – ( type "DINT" )
PosEng – (type "REAL" – units " deg ")
```

PLC source code
```
Pos := TO_DINT(SHL(TO_UDINT(sysDriveStatus[0].wEncAPosHi), 16) OR
sysDriveStatus[0].wEncAPosLo);

PosEng := 360.0 * TO_REAL(Pos) / (65536.0 *65536.0)
```

Note!
Encoder information can be lower than 32bit so user can use only wEncAPosHi to have encoder position.

(2) SPEED SCALE
Motor speed info into drive is calculated as difference of two position at 250us using 32 bit variable:
wSpeedLo is lower word of 32 bit word data;
wSpeedHi is higher word of 32 bit word data.

Cnts2Rpm value inside sysDriveStatus structure shows conversion factor.

Example:

Variable
```
MotorSpeed – ( type "DINT" )
Speed – user read only parameter, (type "REAL", units "rpm")
```

PLC source code
```
MotorSpeed := TO_DINT(SHL(TO_UDINT( sysDriveStatus[0].wSpeedHi ), 16) OR
sysDriveStatus[0].wSpeedLo );

vSpeed := TO_REAL( MotorSpeed ) * sysDriveStatus[0].Cnts2Rpm);
```

(3) CURRENT SCALE
Current scale depends on the drive size.

Cnts2Arms value inside sysDriveStatus structure shows conversion factor.

Example:

Variables
```
MotCur  - user parameter (type “REAL”, unit “Arms”)
```

ST PLC Source Code
```
vMotCur := TO_REAL(sysDriveStatus[0].iIq) * sysDriveStatus[ 0 ].cnts2Arms;
```

(4) VOUT SCALE
Cnts2Vout value inside sysDriveStatus structure shows conversion factor to read Output Voltage.

Example:

Variables

```
OutputVoltage  - user parameter (type “REAL”, unit “Vrms”)
```

ST PLC Source Code
```
vOutputVoltage := TO_REAL(sysDriveStatus[0].iVout) * sysDriveStatus[ 0
].cnts2Vout;
```

(5) MOTOR POSITION SCALE
Motor position scale is similar to encoder. Motor position into drive is normalized at 2^32, so a revolution is 2^32 counts:
wMotPosLo is lower word of 32 bit word data;
wMotPosHi is higher word of 32 bit word data.

Example:

Variable
```
MotPos – ( type “DINT” )
MotPosEng – (type “REAL” – units ” deg ”)
```

PLC source code
```
MotPos := TO_DINT(SHL(TO_UDINT(sysDriveStatus[0].wMotPosHi), 16) OR
sysDriveStatus[0].wMotPosLo);

MotPosEng := 360.0 * TO_REAL(Pos) / (65536.0 *65536.0)
```

NOTE:
Usually motor and encoder are coupled togheter so Encoder Position and motor position are the same.
In some cases where there is a torque motor with high number of pole pairs can be more simple to use an encoder coupled by means a fixed gear ratio. In that case encoder position and motor position are different.

8.5 COUNTERS

Name	Type	Description	Access
sys1kHzTimer	UDINT	System counter with period of 1ms	Read Only
sys4kHzTimer	UDINT	System counter with period of 250us	Read Only

8.6 INPUTS/OUTPUTS

The AXV300 system are supplied with 4 digital inputs, 3 digital outputs, 2 analog inputs and 1 analog outputs located on the standard terminal boards.
The ±10 V analog inputs (0.20mA) have a 12-bit A/D converter. The ±10 V analog outputs (0.50mA) have a 12-bit D/A converter.

The corresponding variables are listed in the following table:

Name	Type	Description	Notes
sysDI_1..sysDI_4	BOOL	Digital inputs	Read Only
sysDI	ARRAY	Digital inputs accessed as array of BOOL	Read Only
sysDO_1..sysDO_3	BOOL	Digital outputs	Read/Write
sysDO	ARRAY	Digital output accessed as array of BOOL	Read/Write
sysAI_1..sysAI_2	BOOL	Analog inputs	12.75V = 32768 [-32678..32767]
sysAI	ARRAY	Analog inputs accessed as array of INT	12.75V = 32768 [-32678..32767]
sysAO_1	BOOL	Analog outputs	12.5V = 32768 [-32678..32767]

8.6.1 sysExtIO

The AXV300 system can manage also external I/O module by means standard CanOpen protocol (refer to **AXV300 User Manual** to detailed description of functionality).

The variables available for the application are listed in the following table

Name	Type	Description	Notes
sysEXTDI_1	UDINT	Digital inputs 0..31	Read Only
sysEXTDI_2	UDINT	Digital inputs 32..63	Read Only
sysEXTDO_1	UDINT	Digital output 0..31	Read/Write
sysEXTDO_2	UDINT	Digital output 32..63	Read/Write
sysEXTAI	ARRAY [0..7] OF INT	Array of external analog inputs	Read
sysEXTAI_1	INT	External Analog input 1	Read
sysEXTAI_2	INT	External Analog input 2	Read
sysEXTAI_3	INT	External Analog input 3	Read
sysEXTAI_4	INT	External Analog input 4	Read
sysEXTAI_5	INT	External Analog input 5	Read
sysEXTAI_6	INT	External Analog input 6	Read
sysEXTAI_7	INT	External Analog input 7	Read
sysEXTAI_8	INT	External Analog input 8	Read
sysEXTAO	ARRAY [0..7] OF INT	Array of external analog outputs	Read/Write
sysEXTAO_1	INT	External Analog outputs 1	Read/Write
sysEXTAO_2	INT	External Analog outputs 2	Read/Write
sysEXTAO_3	INT	External Analog outputs 3	Read/Write
sysEXTAO_4	INT	External Analog outputs 4	Read/Write
sysEXTAO_5	INT	External Analog outputs 5	Read/Write
sysEXTAO_6	INT	External Analog outputs 6	Read/Write
sysEXTAO_7	INT	External Analog outputs 7	Read/Write
sysEXTAO_8	INT	External Analog outputs 8	Read/Write
sysExtIOStatus	UDINT	Status word of external I/O	Read/Write

Scaling of data are defined by manufacturer of connected module.

8.7 AUXILIARY ENCODER

8.7.1 sysAuxEncoder

Name	Type	Description	Access
susAuxEncoder	STRUCT EncoderControl	Auxiliar encoder data structure	Read/Write

Name	Type	Description	Notes	Firmware scale
F0Poscnt	DINT	F0 position capture	Read Only	(1)
F0Rev	DINT	F0 revolution capture	Read Only	--
F1Poscnt	DINT	F1 position capture	Read Only	(1)
F1Rev	DINT	F1 Revolution capture	Read Only	--
CHCPoscnt	DINT	CHC (index) position capture	Read Only	(1)
CHCRev	DINT	CHC (index) revolution	Read Only	--
PosLo	UINT	Encoder postion: low word	Read Only	(1)
PosHi	UINT	Encoder position: hi word	Read Only	(1)
Rev	DINT	Revolutions	Read Only	--
AbsPosLo	UINT	Absolute position low word	Read Only	(1)
AbsPosHi	UINT	Absolute position hi word	Read Only	(1)
AbsRev	DINT	Encoder revolution	Read Only	--
PosCnt	DINT	Position in counts	Read Only	(2)
Virtual_pulses	DINT	Counts per revolutions	Read Only	(3)
Speed	REAL	Filtered encoder speed	Read Only	(4)
errcode	UDINT	Error code	Read Only	
Cnt2Rpm	REAL	Encoder speed scaling factor	Read Only	
Status_Word_F10	WORD	Status Word for capture	Read Only	
Ctrl_Word_F10	WORD	Control word for capture	Read/Write	

(1) Position value is from 0 to 232-1
(2) Encoder option card scales the position in this way:
One revolution is divided in Virtual_pulses steps.
Position value is from 0 to (Virtual_pulses -1).
(3) Virtual_Pulses is calculated by the firmware and is physical encoder pulses * 214.

Example:
Encoder with 2048 pulses per rev then Virtual_Pulses = 2048 * 214 = 33554432.
Position can be from 0 to 33554431.

(4) Speed is filtered, to obtain rpm value it can be used factor Cnt2Rpm.

8.7.2 How to use auxiliary Encoder

Possible Features:

- Read incremental position:

To obtain the 32-bit position is possible to evaluate PosHi and PosLo separately:

```
pos : UDINT;
pos := TO_UDINT(sysAuxEncoder.PosLo) + (TO_UDINT(sysAuxEncoder.PosHi) * 65536);
```

or read it directly through the pointer :

```
ptrPos : @UDINT;
pos : UDINT;
ptrPos := ADR(sysAuxEncoder.PosLo);
pos :=@ptrPos;
```

Is also possible read the position with lower resolution, using only PosHi. In this case 360 ° correspond to 65535.

pos : UINT;
pos := sysAuxEncoder.PosHi ;

The number of revolutions can be read directly in sysAuxEncoder.Rev
Reading sysAuxEncoder.PosCnt in the fast task is possible to read the position encoder count, compared to Virtual_Pulses that represents the number of count of a rotation.

- **Read absolute position: (only encoder SSI ENDAT HIPERFACE)**

The absolute position within one revolution is contained in AbsPosLo and AbsPosHi, with the same mode of PosLo and PosHi. The incremental position is not initialized with the absolute startup.
In the case of multi-turn encoder, the absolute number of revolutions is contained in AbsRev, whose maximum value can be read in parameter 176 "MaxAbsRev".
For Endat and Hiperface encoder and you can send a command to reset the absolute position using parameter 162 "Encoder command" with a Enc SetPos.

- Read filtered speed

The field Speed is only a monitor of internal filtered speed variable. The filter is set by parameter 186 "Encoder spd filter".
To have value in Rpm use Cnts2Rpm factor.

- Calculate encoder speed
To calculate speed between two task executions it ca be used this formula:

Var:
```
Pos : DINT;
DeltaPos : DINT;
PosPrec : DINT;
EncSpdRpm : REAL;
Tp : INT := 1; (* task period , Example = 1 ms *)
```

ST Code:
```
Pos := sysAuxEnc.PosCnt;
DeltaPos := Pos – PosPrec;
EncSpdRpm := TO_REAL(DeltaPos) / ( TO_REAL(VirtualPulses) * Tp ) * 60000.0 ;
PosPrec := Pos;
```

- Error

sysAuxEncoder.errcode is the value of parameter 188 "Encoder error code"
To reset the error using parameter 162 "Encoder command" Encoder Res

- Encoder Capture

This functionality permits to capture the encoder position in very fast way (about 1us).
It is managed from Ctrl_Word_F10 (control word) and Status_Word_F10 (status word).

CNTR_WORD_F10	Name	Description	Notes
Bit 0	CLRFR0	Clear status bit freeze Input F0	Read Only
Bit 1	CLRFR1	Clear status bit freeze Input F1	Read Only
Bit 2	CLRFRC	Clear status bit freeze encoder marker (named C)	Read Only

CNTR_WORD_F10	Name	Description	Notes
Bit 3	--	Not used	Read Only
Bit 4-5	CHF0EDGE	Selection of input F0 edge: 0 : Falling 1 : Rising 2 : Both	Read Only
Bit 6-7	CHF1EDGE	Selection of input F1 edge	Read Only
Bit 8-9	CHFCEDGE	Selection of input C edge	Read Only
Bit 11-10	ENNFR0LT	Enable Capture on F0 trigger: 0 : Disable 1 : Once 2 : Continuos	Read Only
Bit 12-13	ENNFR1LT	Enable Capture on F1 trigger	Read Only
Bit 14-15	ENNFRCLT	Enable Capture on C trigger	Read Only

STATUS_WORD_F10	Name	Description	Access
Bit 0	FR0	Input F0 monitor	Read Only
Bit 1	FR1	Input F1 monitor	Read Only
Bit 2	FRC	Input C monitor	Read Only
Bit 3 -4	STATFR0LT	Status bit of F0 capture module 0 : Disabled 1 : waiting once 2 : Captured once	Read Only
Bit 5-6	CHF0EDGE	Status bit of F0 capture module	Read Only
Bit 7-8	CHF1EDGE	Status bit of F0 capture module	Read Only
Bit 8-9	CHFCEDGE	Selection of input C edge	Read Only
Bit 10-15	--	Not used	Read Only

9 System Embedded Functions

The AXV300 firmware exposes some system embedded functions that can be useful in order to speed-up program execution and to perform some operation with firmware that couldn't be developed with IEC-61131 languages.
These functions are listed in the 'AXV300_IEC' windows inside the MDPlc development environment (Library toolbar).

"ST" invocation from application side	Description	Note
UINT :=sysGetParDInt(UINT,@DINT)	Get integer parameter from Database. **Arguments**: 1- ipa – IPA of parameter 2- Data – pointer to DINT variable to write result **Result**: Return Code from database	Use this function only in Backgroud Task.
UINT := sysGetParInt(UINT,@INT)	Get Integer parameter from Database. **Arguments**: 1- ipa - IPA of parameter 2- data – pointer to INT variable to write result **Result**: Return Code from database	Use this function only in Backgroud Task.
UINT := sysGetParReal(UINT,@REAL)	Get real parameter from Database. **Arguments**: 1- ipa - IPA of parameter 2- data – pointer to REAL variable to write result **Result**: Return Code from database	Use this function only in Backgroud Task.
UINT := sysSetParDInt(UINT,DINT)	Get integer parameter from Database. **Arguments**: 1- ipa – IPA of parameter 2- Data – data value **Result**: Return Code from database	Use this function only in Backgroud Task.
UINT := sysSetParInt(UINT,INT)	Get Integer parameter from Database. **Arguments**: 1- ipa - IPA of parameter 2- Data – data value **Result**: Return Code from database **Note**: Use this function only in Backgroud Task.	Use this function only in Backgroud Task.
UINT := sysSetParReal(UINT,REAL)	Get real parameter from Database. **Arguments**: 1- ipa - IPA of parameter 2- Data – data value **Result**: Return Code from database	Use this function only in Backgroud Task.
BOOL := sysGetDriveObject(UINT,UINT,@UINT)	Get drive object data directly from drive by means GSTAR service channel. **Arguments**: 1- DriveNo – number of drive request 2- Oid – drive object index 3- Data – pointer to variable to write result **Result**: Success of data operation	Use this function only in Backgroud Task.
BOOL := sysSetDriveObject(UINT,UINT,UINT)	Set drive object data directly into drive by means GSTAR service channel. **Arguments**: 1- DriveNo – number of drive request 2- Oid – drive object index 3- Data – data value **Result**: Success of data operation	Use this function only in Backgroud Task
BOOL := sysUpdateDriveObject(UINT,UINT)	Get drive object data directly from drive by means GSTAR service channel and write value directly into drive status var data structure. **Arguments**: 1- DriveNo – number of drive request 2- Oid – drive object index **Result**: Success of data operation	Use this function only in Backgroud Task.

Database embedded function error codes are listed below.

Name	Description
0	No error
1	Non-existing parameter in the database
2	System error
3	Invalid parameter type
4	Read only parameter
5	The parameter cannot be written now
6	Too low value
7	Too high value
8	Internal conflict
9	Non-valid value
10	User function error

Example 1

Variable
Drv0 Max Curr – system parameter IPA 1302
Imax – user local variable (type “REAL”)
Tmp – user local auxiliary variable (type “REAL”)
ADR() – standard operator to get addrees of variable

PLC source code inside Background Task

```
IF ( sysGetParReal( 1302, ADR(Tmp) ) = 0 ) THEN
      Imax := Tmp;
END_IF;
```

Example 2

Variable
Error – flag of error writing (type “BOOL”)
ImaxValue – value to be set (TYPE “REAL”, units ”Arms”)

PLC source code inside Background Task

```
ImaxValue := 2.0;
IF ( sysSetParReal( 1302, ImaxValue ) ) THEN;
      Error := true;
END_IF;
```

Example 3

Variable
VdcLink – user parameter to show drive object data
VdcObjIdx – user costant (type “UINT”) set at value 103

Tmp – user local auxiliary variable (type “REAL”)
ADR() – standard operator to get address of variable

PLC source code inside Background Task

```
IF ( sysGetDriveObject( 0, 103, ADR(Tmp) ) ) THEN
      vVdcLink = Tmp;
END_IF;
```

9.1 FIELDBUS CHANNEL

The Fieldbus data structure sysFbControl is used for any fieldbus channel.

Name	Type	Description	Access
sysFbControl	STRUCT	Fieldbus status	Read Only

Name	Type	Description	Access
ok	BOOL	Fieldbus OK (configuration ok)	Read Only
pdcEnabled	BOOL	Background data exchange Ok (firmware deactive this while PDO configuring)	Read Only
baud	DWORD	Baudrate (equal to parameter IPA 142)	Read Only
State	DWORD	Status (equal to parameter IPA 143)	Read Only
failCause	DWORD	Error code (equal to parameter IPA 145)	Read Only
Addr	WORD	Address (equal to parameter IPA 141)	Read Only
numRecRx	WORD	Number of configured record into PDO M2S	Read Only
numByteRx	WORD	Number of byte used into PDO M2S	Read Only
numParRx	WORD	Number of parameter used into PDO M2S	Read Only
numRecTx	WORD	Number of configured record into PDO S2M	Read Only
numByteTx	WORD	Number of byte used into PDO S2M	Read Only
numParTx	WORD	Number of parameter used into PDO S2M	Read Only

9.1.1 RTE

The RTE data structure sysRteControl is used if configured fieldbus on RTE board.

Name	Type	Description	Access
sysRteControl	WORD	RTE fieldbus status	Read Only

Name	Type	Description	Access
syncFast	BOOL	Enable Fast task synchronization to Fieldbus sync (only for GdNet and Ethercat DC)	Read/Write
syncSlow	BOOL	Enable Slow task synchronization to Fieldbus sync (only for GdNet): do not use	Read/Write
syncOk	BOOL	Fast task syncronized (refresh at 250us)	Read Only
commCycle	WORD	Communication period (us)	Read Only
protocol	WORD	Protocol type: 0 - None 1- Ethetcat 2- Ethernet IP 3- GdNet	Read Only
state	DWORD	0 – Not Operational 1 – Operational	Read Only
version	UINT	Version of RTE protocol software	Read Only

Only sysRteControl.syncFast is writable, use it if needed.

To check data exchange is active, user application has to verify this state
```
sysFbControl.state != 5 AND sysRteControl.state = 1  .
```

To check data exchange loss, user application has to verify this state
```
sysFbControl.state != 5 AND sysRteControl.state = 0  .
```

If Fast task synchronization is enabled (syncFast = TRUE), user application has to check syncOk flag.

Flag behavior in case of communication Ok and communication loss are shown below.

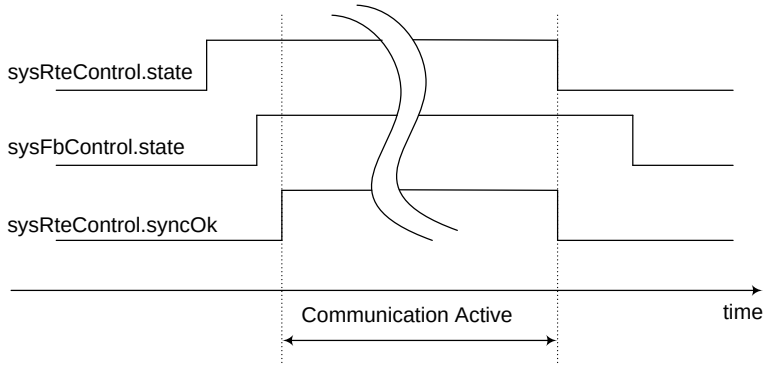


Figura 3 - Fielbus bit state

To set system an alarm user application has to use:
`sysSetPlcAlarm(DWORD dwPlcCode).`

Higher 16 bit of dwPlcCode can used to give system in Alarm state (16 alarm available).
Lower 16 bit are reserved for internal system alarm (so write this lo word has no effect).

9.2 SOFTWARE COMPATIBILITY BETWEEN FIRMWARE VERSIONS

When user recompiles software on new target firmware version, it is useful have a look to following issues to minimize any problem on user application.

9.2.1 AXV300_2_0_1_0 -> AXV300_2_0_3_0

Mandatory
None

Advised
Check new available sysDriveStatus[x].iActVoutLimit
Evaluate that now you can use sysDriveControl[x].iIqRef as Torque feedforward.

9.2.2 AXV300_2_0_0_0 -> AXV300_2_0_1_0

Mandatory
None

Advised
Evaluate if manage pre-alarm of motor overtemperature and if calculate actual motor temperature by mean wMotTemp variable.

9.2.3 AXV300_1_0_3_0 -> AXV300_2_0_0_0

Mandatory
It is needed verify and adapt application menu and user parameters to fit the limitations present:

- Add to any parameters and any variable the field “Short description”
- Verify and, if necessary, modify name of menu and parameters tree

Advised
Evaluate if continue to use the encoder position variables or motor position variables

Evaluate if manage new drive alarms

9.2.4 AXV300_1_0_2_0 -> AXV300_1_0_3_0

Mandatory
Rename sysDriveControl[x].iIqMax to sysDriveControl[x].iPosIqLim.
Manage new sysDriveControl[x].iNegIqLim or set parameter DrvX TrqLim Conf to Sym mode.
Rename sysDriveStatus[x].iIqLim to sysDriveStatus[x].iPosIqLim.

Advised
Evaluate if manage new drive alarms.
Evaluate if manage new sysDriveStatus[x].iNegIqLim.
Evaluate if use GStop firmware function.
Evaluate to use variables of type of RETAIN.
Evaluate the modifications of database embedded function interface (sysDBase)

9.2.5 AXV300_1_0_0_0 -> AXV300_1_0_2_0

Mandatory
None

Faculty
Check new available sysDriveControl[x] variables:
- iVdRef : direct voltage reference
- iVqRef : quadrature voltage reference

Check new available sysDriveStatus[x] variables:
- iIu : current on phase U of the motor
- iIv : current on phase V of the motor

Manage new alarms.

Gefran worldwide

GEFRAN DEUTSCHLAND GMBH
Philipp-Reis-Straße 9a
D-63500 Seligenstadt
Ph. +49 (0) 61828090
Fax +49 (0) 6182809222
vertrieb@gefran.de

SIEI AREG - GERMANY
Gottlieb-Daimler Strasse 17/3
D-74385 - Pleidelsheim
Ph. +49 (0) 7144 897360
Fax +49 (0) 7144 8973697
info@sieiareg.de

SENSORMATE AG
Steigweg 8,
CH-8355 Aadorf, Switzerland
Ph. +41(0)52-2421818
Fax +41(0)52-3661884
http://www.sensormate.ch

GEFRAN FRANCE SA
4, rue Jean Desparmet - BP 8237
69355 LYON Cedex 08
Ph. +33 (0) 478770300
Fax +33 (0) 478770320
commercial@gefran.fr

GEFRAN BENELUX NV
ENA 23 Zone 3, nr. 3910
Lammerdries-Zuid 14A
B-2250 OLEN
Ph. +32 (0) 14248181
Fax +32 (0) 14248180
info@gefran.be

GEFRAN UK LTD
Unit 7, Brook Business Centre
54a Cowley Mill Road, Uxbridge,
UB8 2FX
Ph. +44 (0) 8452 604555
Fax +44 (0) 8452 604556
sales@gefran.co.uk

GEFRAN MIDDLE EAST ELEKTRIK VE ELEKTRONIK SAN. VE TIC. LTD. STI
Yesilkoy Mah. Ataturk
Cad. No: 12/1 B1 Blok K:12
D: 389 Bakirkoy /Istanbul
TURKIYE
Ph. +90212 465 91 21
Fax +90212 465 91 22

GEFRAN SIEI
Drives Technology Co., Ltd
No. 1285, Beihe Road, Jiading
District, Shanghai, China 201807
Ph. +86 21 69169898
Fax +86 21 69169333
info@gefran.com.cn

GEFRAN SIEI - ASIA
31 Ubi Road 1
#02-07, Aztech Building,
Singapore 408694
Ph. +65 6 8418300
Fax +65 6 7428300
info@gefran.com.sg

GEFRAN INDIA
Survey No. 191/A/1,
Chinchwad Station Road,
Chinchwad,
Pune-411033, Maharashtra
Ph. +91 20 6614 6500
Fax +91 20 6614 6501
gefran.india@gefran.in

GEFRAN INC.
8 Lowell Avenue
WINCHESTER - MA 01890
Toll Free 1-888-888-4474
Fax +1 (781) 7291468
info.us@gefran.com

GEFRAN BRASIL
ELETROELETRÔNICA
Avenida Dr. Altino Arantes,
377 Vila Clementino
04042-032 SÂO PAULO - SP
Ph. +55 (0) 1155851133
Fax +55 (0) 1132974012
comercial@gefran.com.br

GEFRAN

GEFRAN S.p.A.
Via Sebina 74
25050 Provaglio d'Iseo (BS) ITALY
Ph. +39 030 98881
Fax +39 030 9839063
info@gefran.com
www.gefran.com

Drive & Motion Control Unit
Via Carducci 24
21040 Gerenzano [VA] ITALY
Ph. +39 02 967601
Fax +39 02 9682653
infomotion@gefran.com

Technical Assistance :
technohelp@gefran.com

Customer Service :
motioncustomer@gefran.com
Ph. +39 02 96760500
Fax +39 02 96760278