Quick Guide for writing applications with MDPlc tool
This manual applies to the hardware and software configurations of the following drives:

ADV200: software version V 7.x.x
APC300: software version V 1.x.x

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: technohelp@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 Spa 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.
All rights reserved
TABLE OF CONTENTS

1 INTRODUCTION ........................................................................................................................................ 5
  1.1 PURPOSE OF THIS GUIDE ............................................................................................................... 5
  1.2 MDPLC ................................................................................................................................................ 5
  1.3 CONTENTS OF THE GUIDE .............................................................................................................. 5

2 THE IEC 61131-3 STANDARD .................................................................................................................. 6
  2.1 SHORT DESCRIPTION OF IEC 61131-3 STANDARD ...................................................................... 6
  2.2 COMMON ELEMENTS ....................................................................................................................... 6
  2.3 THE LANGUAGES .............................................................................................................................. 6

3 THE MDPLC ENVIRONMENT ................................................................................................................. 10
  3.1 THE RUN-TIME ENVIRONMENT ..................................................................................................... 10

4 MATERIAL REQUIRED ........................................................................................................................... 13

5 INSTALLATION ....................................................................................................................................... 14
  5.1 GF_eXpress SETUP PROCEDURE ................................................................................................. 14
  5.2 ADV200 SUPPORT FILE SETUP PROCEDURE ............................................................................. 20
  5.3 APC300 SUPPORT FILE SETUP PROCEDURE ............................................................................. 22
      5.3.1 APC300 Gf-eXpress CATALOG PROCEDURE ........................................................................ 25
  5.4 MDPLC SETUP PROCEDURE ......................................................................................................... 27
  5.5 ADV200 MDPLC LIBRARY SETUP PROCEDURE .......................................................................... 28
  5.6 THE MDPLC DIRECTORY ............................................................................................................... 30

6 MAIN ELEMENTS OF MDPLC ................................................................................................................ 31
  6.1 MDPLC STRUCTURE ...................................................................................................................... 31
  6.2 PROJECT WINDOW ......................................................................................................................... 32
  6.3 EDITOR OF THE VARIABLES ......................................................................................................... 32
  6.4 PROGRAM EDITOR ............................................................................................................................ 33
  6.5 WATCH WINDOW .............................................................................................................................. 33
  6.6 OUTPUT WINDOW ........................................................................................................................... 34
  6.7 LIBRARY WINDOW .......................................................................................................................... 34
  6.8 TRIGGER WINDOW ......................................................................................................................... 35

7 HOW TO CREATE A SIMPLE APPLICATION ....................................................................................... 38
  7.1 PURPOSE ......................................................................................................................................... 38
  7.2 PRELIMINARY OPERATIONS ......................................................................................................... 38
  7.3 CREATION OF THE PROJECT ......................................................................................................... 39
  7.4 CREATION OF THE PROGRAM ....................................................................................................... 40
  7.5 ASSIGNMENT OF A PROGRAM TO A TASK .................................................................................. 41
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>7.6</td>
<td>CONNECTION TO THE TARGET</td>
<td>42</td>
</tr>
<tr>
<td>7.7</td>
<td>COMPILATION OF THE CODE AND DOWNLOAD</td>
<td>42</td>
</tr>
<tr>
<td>7.8</td>
<td>FIRST STEPS IN DEBUGGING</td>
<td>43</td>
</tr>
<tr>
<td>7.9</td>
<td>INTERFACING WITH THE VARIABLES OF THE TARGET SYSTEM</td>
<td>45</td>
</tr>
<tr>
<td>7.10</td>
<td>CREATION OF THE PARAMETERS</td>
<td>46</td>
</tr>
<tr>
<td>7.11</td>
<td>USE OF THE PARAMETERS IN THE APPLICATIONS</td>
<td>49</td>
</tr>
<tr>
<td>8</td>
<td>OTHER PROGRAMMING OPTIONS</td>
<td>52</td>
</tr>
<tr>
<td>9</td>
<td>SOFTSCOPE</td>
<td>54</td>
</tr>
</tbody>
</table>
1 INTRODUCTION

1.1 PURPOSE OF THIS GUIDE

This document is a basic guide intended to introduce the use of the MdPlc development environment. The guide illustrates the basic steps (from program setup to active application on the drive) to be carried out by the user in order to acquire basic knowledge of this tool.

More detailed information can be found on-line in the help window generated by installation of MdPlc. To activate the on-line help window, press the F1 key in the MdPlc environment.

The programming examples indicated in the guide refer to the ADV200 drive.

1.2 MDPLC

MdPlc is a development environment based on IEC 61131-3 PLC standard languages.

With MdPlc, the programmer can write PLC applications for Xvy servodrive, ADV200 drive and APC300 card for TPD32-EV DC converter using all five different languages provided by the IEC standard.

MdPlc also features debug capabilities which simplify application testing.

1.3 CONTENTS OF THE GUIDE

The path followed by this guide has been divided as follows:

- Short description of the IEC 61131-3 standard and of its five languages;
- Short introduction to the MdPlc environment;
- Preparation of the program;
- Main elements of MdPlc;
- Development of a simple application.
2 THE IEC 61131-3 STANDARD

2.1 SHORT DESCRIPTION OF IEC 61131-3 STANDARD

IEC 61131-3 is a standard industrial automation programming language. The standard is supported at International level and is therefore independent. It is divided mainly into two groups:

- Common elements;
- Programming languages.

The MdPlc help window, available on-line, contains a reference section with detailed information on the IEC 61131-3 standard.

2.2 COMMON ELEMENTS

The common elements of the standard refer to the following topics:

- Definition of type of data;
- Declaration of the variable;
- Configuration of the task;
- Definition of programming units (programs, functions and functional blocks).

2.3 THE LANGUAGES

Four programming languages are defined in the standard. This means that definition of the syntax and semantics has to strictly followed.

The languages consist of two textual versions and two graphic versions:

Textual version:

- Instruction List IL
- Structured Text ST

Graphic version:

- Ladder Diagram LD
- Function Block Diagram FBD

A fifth language, the Sequential Function Chart language also called SFC, has also been defined. SFC is not really a language but, on the contrary, provides a formal structure for representation of the sequences of a control program.
Instruction List (IL)

The Instruction list language resembles programming of the Assembler language. The instructions are organized in a list with one instruction per line. Each instruction acts on the accumulator register which contains the current result.

IL is ideal for solving minor problems with few decision-making points and a limited number of changes in the execution flow of the program.

Structured Text (ST)

The Structured Text language is a powerful, high level language similar to Pascal or Basic. It contains all the main elements of a modern programming language, including assessment sections (IF-THEN-ELSE and CASE OF) and iteration loops (FOR, WHILE and REPEAT). These elements may also be nested in a multilevel language structure.

It is an excellent tool for defining complex functional blocks that can be used inside any of the other languages.
Function Block Diagram (FBD)
The Function Block Diagram language is frequently used in industrial processes. It tracks the behavior of the functions, of the functional blocks and programs as a group of interlinked graphic blocks similar to electronic circuit diagrams. It views a system in terms of flow of the signals between the elements in the processing phase.

Ladder Diagram (LD)
Representation of logical sequences with the LD language derives from traditional relay-based logic design in the electrical system engineering environment.
LD is particularly suited for operations on digital signals and on Boolean variables.
Sequential Function Chart (SFC)
SFC programming provides a graphic method of program organization. The three main components of the SFC language are sequences, actions and transitions. The sequences account for most of the logic and are, therefore, a unit of the programming logic that performs a particular control task. The actions are the individual aspects of the specific task. The transitions represent the mechanism used to move from one task to another. The control logic for each Sequence, Action and Transition is programmed in one of the other languages, such as for example Ladder Diagram or Structured Text.
3 THE MDPLC ENVIRONMENT

3.1 THE RUN-TIME ENVIRONMENT

A short presentation of the MDPlc system is provided below.

- The MdPlc project comprises all the source modules, such as Programs, Functions and Functional Blocks created with one of the PLC languages supported by MdPlc. The parameters of the application are also defined in the project.
- The MDPlc compiler allows the user to check projects, to edit source modules and the parameter table and to assign each single module to one of the executive tasks of the drive.
- MDPlc communication functions make it possible to send the compiled application code to the drive on the communication interface.
- During execution of the application code, the value of the variables can be read and the execution flow of the PLC program can be checked using the debug tools provided by MDPlc.
The GF_eXpress configurator allows the user to carry out the following tasks: parameterization of the application downloaded in the drive of the target, supervision of drive activities, diagnostics and service functions. To carry out these functions, the GF_eXpress configurator uses the parameter files produced by MDPLc during compilation.

NOTE:
Starting from ADV200 Fw V4.0.0 GF_eXpress must be used not as an alternative but in replacement of E@syDrive. The parameter files generated by MDPLC in old format *.par, must be converted to the *.Gfs, this is possible using a special GF-eXpress features.

Convert MDPLC par file

From GF_eXpress:

1: [Image]
2: [Image]

This tool converts a ".par" file created by MDPLC.
This program has the following input boxes:
- Parameter File : path of the input file (.par)
- Configuration File : path of the configuration file (AppConfig.xml)
- gfe destination folder: path of the folder containing the .gfe output file (optional)

It builds a .xml file extended as .gfe (Output File) starting from a .par file, and a .gft file of the MDPLC application. Configuration File path is already set.

If you want to distribute a MDPLC application parameter file in GF_eXpress it's necessary to Export and Import the Configuration.
The complete procedure is indicated in the GF_eXpress Manual, here the most important informations:

Import Export configuration
With the “Import Configuration” and “Export Configuration” commands on the “File” menu, you can import and export GFE files with assigned GFT in a single ZIP packet. This function lets you reuse the configuration and the definition of a specific custom device created by the user.

It is used to export to another PC a device not included in the standard catalog, typically a device written with MDPLC program.

This window tool has the following input boxes:

- Gfe FileName: path of the input file (.gfe)
- Output Folder: path of the folder containing the .zip output file

It builds a .zip file containing the .gfe file and the relative .gft file, inclusive of all over folders until Catalog\.

For example, if the saved file.gfe refers to .gft file ADV200_5_X_0.gft, located in C:\Programmi\Gefran\Catalog\Custom\App\ADV200_5_X_0, then the program will generate a .zip file named saved file.zip, containing the file saved.gfe and the folder Custom\App\ADV200_5_X_0\ADV200_5_X_0.gft

This window tool has the following input boxes:

- Zip FileName: path of the input file (.zip), already made by exportation option
- Output Folder: path of the .gfe file just extracted

It unzips the .zip file checked, then places the .gft file in the original position and saves the .gfe file in the desired path.

All the existing files will be overwritten.
4 MATERIAL REQUIRED

Programming and development software:

- Mdplc installation disk version 5.98.9.6 or later
- GF_eXpress installation file version 1.7.0 or later
- GF_eXpress catalog installation file version 2.2.0 or later
- ADV200 installation disk version 6.0.0 or later for ADV200 drive
- APC300 installation disk version 1.0.0 or later for APC300 Board

Connection to a PC with RS232 port:
The following are required for connection:

- an optional PCI-COM (or PCI-485) adapter, code S560T.
- shielded cable for the XS / PCI-COM (or PCI-485) connection, code 8S8F59, see figure 5.5.1.1.

Connection to a PC with USB port:
The following are required for connection:

- an optional PCI-COM (or PCI-485) adapter, code S560T.
- an optional USB/RS232 adapter, code S5A20 (including the cable for USB connection)
- shielded cable for the XS / PCI-COM (or PCI-485) connection, code 8S8F59, see figure 5.5.1.1.
5 INSTALLATION
The installation must be preceded by the installation of GF_eXpress Catalog. All the following information is contained in the manual "GF_eXpress user guide" downloaded from Gefran website.

5.1 GF_eXpress SETUP PROCEDURE
GF_eXpress installation must be preceded by the installation of the program Catalog.

Installation of Catalog
Follow the steps below to install Catalog:
- Execute GF_Catalog_version#.exe and the welcome screen appears:

  Welcome to the Catalog Setup Wizard
  This will install Catalog 1.7.0 on your computer.
  It is recommended that you close all other applications before continuing.
  Click Next to continue, or Cancel to exit Setup.

  • Click Next to proceed to the next screen

  License Agreement
  Please read the following important information before continuing.

  GEFRAN SPA LICENSE AGREEMENT
  IMPORTANT: PLEASE READ THE TERMS AND CONDITIONS OF THIS LICENSE AGREEMENT CAREFULLY BEFORE USING THE SUPPLIED SOFTWARE
  This License Agreement is a legal agreement between you (either an

  • Read the license agreement, choose I accept the agreement and click Next to proceed
• Select the installation folder and click **Next to proceed**

• Check the settings and click **Install** to proceed; the extract phase starts

• At the end of the extract phase the last window appears:
• Click Finish to close the setup program

Installation of GF_eXpress and RS-USBX driver

Follow the steps below to install GF_eXpress::

• If you need to install the RS-USBX interface make sure to disconnect it from USB port
  (not used for MDPLC ADV200, XXY-EV & APC300 card but only for other Gefran products)
• Execute GF_eXpress_#version#.exe and the welcome screen appears:

  • Click Next to proceed to the next screen
• Read the license agreement, choose *I accept the agreement* and click *Next* to proceed.

• Select the installation folder and click *Next* to proceed.
• Select the Start menu folder and click Next to proceed

![Screen shot of selecting additional tasks](image1)

• Select the additional tasks you want to install (for example drivers for RSUSBX) and click Next to proceed

![Screen shot of ready to install](image2)

• Check the settings and click Install to proceed; the extract phase starts

![Screen shot of installing](image3)
• Only when install drivers for RS-USBX: the following message may appear on PCs running Windows XP, when setup program begins to install the USB drivers:

![Installazione hardware](image)

Il software che si sta installando per l'hardware:
Axel Service Port

non ha superato il testing del programma Windows Logo che consente di verificare la compatibilità con Windows XP. (Informazioni sul testing)

L'installazione del software potrebbe impedire il corretto funzionamento del sistema o renderlo instabile. Microsoft consiglia di arrestare l'installazione e di contattare il fornitore dell'hardware per ottenere un prodotto software che abbia superato il testing del programma Windows Logo.

![Continua](image)

• Ignore this message and continue with installation
• At the end of the extract phase the last window appears

![Completing the GF_eXpress Setup Wizard](image)

Completing the GF_eXpress Setup Wizard

Setup has finished installing GF_eXpress on your computer. The application may be launched by selecting the installed icons.

Click Finish to exit Setup:

- Launch GF_eXpress

![Finish](image)

• Click Finish to close the setup program
5.2 ADV200 SUPPORT FILE SETUP PROCEDURE

CD installation setup process starts showing the following windows:

Select Adv200AsyPlc 6_0_0 to install the files related to version V6_0_0 for asynchronous motor or select the correct fw. Version that you want to install and use in the drive.

Select Adv200SynPlc 6_0_0 to install the files related to version V6_0_0 for brushless synchronous motor

If you install directly from file, run the "setup" from the "Adv200AsyPlc 6_0_0" or "Adv200SynPlc 6_0_0" Sequence for running the setup of the support files:

1) Exit from the Windows programs before running this Setup program.
2) Start the SETUP.EXE file from CD. To do this, open Explorer, move to CD-ROM, double-click on the SETUP.EXE file and follow the instructions. Some of the windows displayed during the installation procedure are shown below (ex. For V4_0_0).
3) At this point, it is possible to select any directory and also the Program folder. It is suggested not to change the directory in order to facilitate update of all ADV200 drive software.

![Select Destination Directory]

4) If you are installing a new version, refer to the revisions log.

![Completing the ADV200Pc 4_0_0 Setup Wizard]

5) The installation procedure adds the “SIEI PC TOOLS” section, in which all the software of the ADV200 drive is installed (by default), to the Windows program menu.

![Windows Program Menu]
To remove the ADV200 software, click on the uninstall icon present in this folder.

5.3 APC300 SUPPORT FILE SETUP PROCEDURE

APC300 CD setup process starts:

Select 5-MdplcApc1.0.0 to install the files related to version V1_0_0.
If you install directly from file, run the "setup" from the "5-MdplcApc1.0.0" Sequence for running the setup of the support files:

6) Exit from the Windows programs before running this Setup program.

7) Start the SETUP.EXE file from CD. To do this, open Explorer, move to CD-ROM, double-click on the SETUP.EXE file and follow the instructions. Some of the windows displayed during the installation procedure are shown below (ex. For V1_0_0).
8) At this point, it is possible to select any directory and also the Program folder. It is advisable not to change the directory in order to facilitate update future software revision.
Press Next:

![Image of installation prompt]

Press Install:

After the installation:

9) If you are installing a new version, refer to the revisions log.

![Image of installation wizard]

10) The installation procedure adds the “SIEI PC TOOLS” section, in which all the software of the APC300 drive is installed (by default), to the Windows program menu.
Now if you want to remove the APC300 software, click on the uninstall icon present in this folder.

5.3.1  APC300 Gf-eXpress CATALOG PROCEDURE

From the APC300 setup:

Select 3-Apc300 1.0.0 to install the GF-eXpress catalog files related to version V1_0_0
Press Next:

![Setup - GF_eXpress APC300 1.0.0](image)

Next:

![Setup - GF_eXpress APC300 1.0.0](image)

Press Install, start the installation. After the installation you can see the following:

![Setup - GF_eXpress APC300 1.0.0](image)

Finish.
5.4 MDPLC SETUP PROCEDURE

MdPlc setup execution sequence:

1) Exit from the Windows programs before running this Setup program.

2) Start the SETUP.EXE file from CD. To do this, open Explorer, move to CD-ROM, double-click on the SETUP.EXE file and follow the instructions. Some of the windows displayed during the installation procedure are shown below.

3) At this point, any directory can be selected as Program folder. It is suggested not to change the directory in order to facilitate update of all drive software.

4) If you are installing a new version, consult the revisions log.
5) The installation procedure adds the “SIEI PC TOOLS” section, in which all the software of the Target (es.ADV200) are installed (by default), to the Windows program menu.

6) To remove the Target (es.ADV200) installation, click on the uninstall icon present in this folder.

5.5 ADV200 MDPLC LIBRARY SETUP PROCEDURE

Run the setup library MDPLC for ADV200 drive.
Then setup program must be executed:

Then setup program must be executed:

Then setup program must be executed:

At this point you can select any directory as the folder of the program. It is recommended not to change the directory in order to more easily update all the software ADV200.

After installation you will have:

Even for libraries, the installation procedure adds to the “Siei Pc Tools” directory all files, including the possibility to uninstall the libraries.
5.6 THE MDPLC DIRECTORY

At the end of the setup phase, the structure of the MdPlc directory is shown in the image below:

The contents of each folder are described below:

7) MdPlc2: is the main folder and contains the executable file of MdPlc, the DLL of the program and various configuration files.

8) Libraries: contains the libraries of the IEC blocks (with .pli extension) to be used and imported in MdPlc projects.

9) Projects: is the default folder containing the MdPlc applications. “Speed” is a basic sample application available with the setup procedure.

10) Service: this folder contains the firmware upgrades for the drive and the parameter files for drive configuration and setup.

11) Targets: this folder contains the target definition files. The targets are various GEFRAN products (XVy-EV, ADV200, APC300 etc.) and the related software versions that can be programmed with MdPlc.

IMPORTANT: The contents of the folders indicated above must never be modified or cancelled manually.
6 MAIN ELEMENTS OF MDPLC

6.1 MDPLC STRUCTURE

The MdPlc compiler during execution of a PLC project is shown in the figure below. When MDPLC program is installed for the first time, not all the windows and tool bars shown in the figure below are visible, they can be selected by using the options of the “View/ToolBars” menu. The MdPlc environment is designed to facilitate editing and debugging of the code. Almost all project elements can be managed in drag & drop mode in various windows.
6.2 PROJECT WINDOW

The project window is usually positioned to the left of the MdPlc window. The project window consists of three folders with the following names:

- **Project**: contains the elements of the main project, such as for example all the blocks of code (program, function and functional blocks) also called POU (Program Object Unit), the variables of the global applications, the variables associated with the parameters of the application, the definitions of the tasks and the configuration.

- **Parameters**: contains the definitions of the parameters of the applications with the definitions of the menu, the descriptions of the alarms, enumerative events and other elements used to define the parameter set of the drive.

- **Macros**: the macros are parameterizable blocks of code that can be used in the project POU. They are useful for creating optimized blocks of code as they are run without a call to the function, thus saving time.

6.3 EDITOR OF THE VARIABLES

Each POU (block of code) has its own editor for local variables. An independent editor is available for global variables. The variables editor makes it possible to add, remove, copy and paste the records of the definitions of the variables. Each record has fields referring to the type of data, address, descriptions of the dimensioning of the arrays etc. Classification of the columns is possible for most of the fields. The global variables editor also features the “Group” characteristic which makes it possible to create like groups of variables. The groups of variables are represented in the project structure in separate folders in order to facilitate management. The variables can be easily moved between the editors and the code and debug windows.
6.4 PROGRAM EDITOR

The program editor is available in the five different versions required by the respective languages of the IEC 61131-3 standard.

In the description of the languages, it is possible to view various screen pages of the code editor (see paragraph 2.3)

All the code editors support drag & drop, cut+copy+paste and an unlimited number of cancel-restore functions also for the graphic editors.

6.5 WATCH WINDOW

Two different watch windows are available for checking of the variables:

- **Textual watch window**: the variables can be inserted in the window by the code and variable editors.

  The values are read by the drive and constantly updated.

  The numeric format (decimal, hexadecimal or floating point) can be modified individually for each single variable.

  The value of the variables can also be forced using a specific option on the tool bar.

- **Graphic watch window**: a maximum of 8 tracks can be inserted in this window.
The list of tracks indicates, individually, the scale and minimum, maximum values. Acquisition of the data can also be saved in text files for subsequent analysis.

### 6.6 OUTPUT WINDOW

The output window shows all the messages of the compiler concerning loading of the project, compilation and downloading of the code. A second folder of the output window is used to list the result of the “find in project” search. Double-clicking on the error message of the compiler or on the result of a search, the specific code editor will be opened automatically and the related text will then be selected.

![Output Window](image)

### 6.7 LIBRARY WINDOW

The library window groups together the following elements of a project:

- IEC standard operators;
- Target variables;
- Integrated blocks;
- Blocks of the library.

The **IEC standard operators** are listed in the first folder. These are the basic blocks for writing the code. The standard blocks are simple operators such as arithmetic blocks, comparators, assignments, skips, etc.

The **target variables** form the data interface with the firmware of the drive. The list of these variables is provided by GEFRAN with the definition and support files for each individual drive. Creation of a new project using MdPlc establishes an automatic link between the specific group of target variables and the project.
The integrated blocks, if available, are functions and functional blocks published by the firmware of the drive. These also come from the GEFRAN firmware support files.

The library folder groups together the blocks arising from a single library linked to the current MDPlc project. The user can select the correct group of the library according to project requirements. The libraries are provided by GEFRAN or can be created by the user using the specific MDPlc library functions.

6.8 TRIGGER WINDOW

The trigger windows are similar to the watch windows described in paragraph 6.5, but are used for real-time deterministic debugging. They are a very powerful tool for analysis of the code.

Unlike the watch windows, the trigger windows refer to a single execution point in the source code selected by the user.

The trigger windows show the value assumed by the variables selected in the position selected.

Sampling tools (expression of the trigger condition, single sequence acquisition, setting of the trigger value in count, etc.) are available so that the user can use the desired program setting.

The value is sampled directly by the drive.

The trigger windows are also available in two versions: textual trigger and graphic trigger.

- Textual trigger

Up to 16 textual triggers can be activated at 16 different execution points of the code.

These are useful for determining the execution flow of the code and for understanding the value of the same variable at the various execution points.

Another important information item provided by the textual triggers is the count of execution of the code at the point selected.
The graphic trigger can be used once time only in the entire code of the project. It records all the values of maximum four variables at the execution point selected. The samples are held in the memory of the drive and, at the end of acquisition, are sent to MdPlc. The graphic trigger is very useful for studying the behavior of the variables in time. The result of acquisition may be saved in a text file for subsequent analysis.
7 HOW TO CREATE A SIMPLE APPLICATION

7.1 PURPOSE

This chapter describes the main steps in creating a simple MdPlc application for the ADV200 drive. The same steps can also be carried out to create applications for other GEFRAN targets that support MdPlc (for example APC300 board).

The sample application described in this chapter simply manages a motor with speed loop control. In the following instructions, all the user commands for MdPlc are indicated with the options of the menu (indicated in boldface). Most MdPlc commands are also available on the tool bar. Some of these can also be carried out using the fast keys.

7.2 PRELIMINARY OPERATIONS

1) If not already carried out, perform the setup of MdPlc as indicated in chapter 5.
2) Install Gefran catalog and GF_eXpress configurator program on the PC.
3) Make sure that communication between the drive and GF_eXpress is working correctly. Refer to the user manual of the ADV200 drive.

With a factory-configured drive, communication must be set as follows:

Communication/Setting menu
4) The recommended programming reference for ADV200 is this document (“Quick Guide for writing applications with MDPlc”). This instrument is useful for understanding the meaning and behavior of the target variables.

5) To carry out the steps indicated in paragraph 7.9, the drive must be wired to a motor. The drive must therefore be suitably configured (motor and encoder characteristics, limits, etc.). The motor should be able to rotate freely without any connection of mechanical parts. The enable switch must also be connected to the drive as indicated to the drive manual.

7.3 CREATION OF THE PROJECT

1) Open MdPlc from the Windows start menu.
   - From the “File” menu, select “New project”; the specific dialogue box will be displayed.

2) Select the directory in which the support files have been installed as destination directory. In this case, \SIEI PC tools\E@syDrives\ADV200_1_X_0 and enter “Test00” as name of the project.
   - Use “Manual project settings”.

   - In the Project menu, access the Options window and set Read target parameters to ON.
3) Select the ADV200 target. Available targets depend on GEFRAN installations. In this case, the ADV200 1.0.0 version of ADV200 is used.

4) Press the OK key. MdPlc starts processing. The output window will show the steps referring to loading of the project. The basic operators and target variables will be inserted in the library window, if visible.

5) At the end of project loading, the output window should show the final message “0 warnings, 0 errors”. Otherwise, check that the ADV200 support files have been installed correctly on the PC.

7.4 CREATION OF THE PROGRAM

1) Select the “Project” + “New object” menu option or right-click on the structure of the project in the project window.

2) The “New object” dialogue box will be opened. This window will be displayed each time a new block of code is to be created. It also makes it possible to create global variables.

3) Select “Program” for POU type, “ST” for the language and “Slow” type as name of the new block of code.

4) Press the OK key. A new editor of the ST code will be opened automatically with the cursor positioned on the first line of the editor.
5) We will now define a local variable in the new Slow program. To do this, use the “Variables” + “Insert” option. An empty variable record will be displayed in the editor of the local variables.

6) Click on the “Name” field, enter the name of the “counter” variable and then press Enter. Click on the “Type” field and enter INT.

7) Click on the source code editor and enter “counter := counter + 1;”. The editor window should be displayed as follows.

8) This very simple program illustrates the main steps in writing MdPlc code. It is also a fast method for checking correct system configuration and functioning. The following steps complete the sequence of the program with compilation and download.

7.5 ASSIGNMENT OF A PROGRAM TO A TASK

1) The programs of the projects must be assigned to a task of the drive so as to manage these in the period of time required by the system firmware of the drive.

2) If not already visible, open the structure of the project in the Project Window (see par. 6.2).

3) Double-click on the “Tasks” icon displayed in the lower part of the project structure. At this point, the task configuration dialogue box will be opened. Available system tasks are displayed together with their execution time (if applicable) and the ID/priority of each task.

4) Click on the “Program” field of the “Slow” task.

5) Edit the field and enter the name “Slow”, which is the name of the program that is to be assigned to the “Slow” task.

6) As an alternative to direct entry of the name, it is possible to press the “Select” key. The “Object browser” window will be opened containing a list of all the programs defined in the project.

7) Select the program and confirm with OK.

8) The “Task configuration” window should appear as in the figure to the side. Press OK to confirm assignment of the task.
7.6 CONNECTION TO THE TARGET

1) Before compiling the first code, it must be possible to communicate with the drive. After initial compilation of the project, it will no longer be necessary to work on line. This operation is necessary as MdPlc must receive the map of the addresses from the drive.

2) The MdPlc connection procedure to the drive is similar to that of GF_eXpress. Selecting the “Communication” + “Settings” option, the communication dialogue box will be displayed.

3) Take note of the communication parameters of GF_eXpress such as protocol () and the address of the drive. Then, apply the same parameters in the MdPlc dialogue boxes. The figures show the dialogue boxes for MdPlc configuration.

4) After suitably setting the communication parameters, press OK twice. The settings will be saved with the project.

5) At this point, select the option of the “Communication” + “Connect” menu to establish communication with the drive. Check that communication is working correctly: the indicator of the status bar should be green, as shown below.

6) If the red error string is displayed, check the hardware connection and/or software settings.

7.7 COMPILATION OF THE CODE AND DOWNLOAD

1) For compilation of the project, press “Project” + “Compile”. The compilation process starts; the operations of the compiler are indicated in the output window. If there are no programming errors, the final message should be “0 warnings, 0 errors” as indicated below. Statistics referring to generation of the code (size of the code, free spaces, etc.) are also displayed.
2) In the case of compilation errors, the related error messages are shown in the output window. Double-clicking on the error message, the incorrect source code will be automatically selected in the editor.

3) The compilation process also generates the parameter file with the name of the project and .PAR extension. In this example, the name will be “Test00.par”. Convert the file in GFS format.

4) Open the parameter file with GF_eXpress and check that the parameters:
   - 554 Access Mode is set to “Expert”
   - 558 Application Select is equal to “Application 1”

5) Download the code with the option of the “Communication” + “Download” menu. A progress bar will indicate the stage of completion of the code download procedure. The output window will display the details of download operations. The drive will reset automatically at the end of downloading.

### 7.8 FIRST STEPS IN DEBUGGING

1) Open the watch window with the command of the “View” + “Watch bar” menu. The bar opens normally to the right of the MdPlc window.

2) Check that communication is still active (see the status bar indicator).

3) Select the “counter” word in the code editor of the “Slow” program and move this to the watch window.
4) The value of the variable should change quickly in the watch window. This means that the PLC code is active on the drive and that this simple application is running correctly.
7.9 INTERFACE WITH THE VARIABLES OF THE TARGET SYSTEM

So far, we have provided a general overview of management of the MdPlc project. The following steps will refer to specific programming of the drive.

The application we are going to write must set the ramp reference to a fixed value or from analogue input according to the status of a digital input.

We will now see how the program we are writing, called ‘application’ below, and the basic firmware are interfaced.

1) The code we are going to write is as follows:

```
counter := counter + 1;
IF (sysDIBitWordBit1)
THEN
    sysRampRef := sysAI0;
ELSE
    sysRampRef := 1000;
END_IF;
```

According to the value of `sysDIBitWordBit1` (status digital input 1), the IF THEN ELSE instruction assigns `sysRampRef := sysAI0;` (if `sysDIBitWordBit1` is high). Otherwise, `sysRampRef := 1000;`.

We will now see how to construct this code.

Before proceeding, check that the conditions indicated in paragraph 7.2 are met.

2) If not yet visible, open the “Library Window” (see paragraph 6.7) with “View” + “Library”.

3) Select the panel of “Target variables” of the “Library Window”. The complete list of target variables will be displayed. To display the details of these variables, right-click on the list and select “View details” in the pop-up menu.

Note: to find the variables available on the drive, it may be handy to sort these in groups. To do this, press the right-hand key of the mouse in the library window. A menu will be displayed in which to select display mode.

4) Write the code copying this from the example above. To trace the variables, it is possible:
to search for the “sysRampRef” variable in the library window and select this.

Retrieve this with drag and drop from the “Library Window” and position it in the editor of the code of the “Slow” program.

Alternatively, it is possible to enter “sysRampRef” in the code editor. Comply with the format of the characters (upper/lower case) as the MdPlc compiler is case sensitive.

Note: “:=” is the assignment instruction
  “=” is the compare instruction
  the instructions end with “;”.

5) Addition of comments is not required but is advisable.

6) The window of the editor should appear as in the figure below. The test variables have already been dragged into the debug window.

7) Compile and download the project carrying out the steps indicated in paragraph 7.7.

8) Activate and deactivate digital input 1. The value sysDIBitWordBit1 in the watch window must change similarly to the other values. The status of the variables can be checked by shifting these from the code editor or from the “Library Window” into the “Watch Window”.

7.10 CREATION OF THE PARAMETERS

We will now pad out our application. Instead of using a fixed value, we want to use a parameter. Also, we want to read the counter value from the keyboard.

To do this, we will use parameters.
Creation of the parameters of the application that can be used to exchange values between the PLC application and the keypad, the GF_eXpress configurator or the field bus is described below.

The PLC application code can act on the parameters using an associated variable created automatically by MdPlc.

The parameters are also saved in the flash memory of the drive in order to preserve these values for subsequent drive re-start phases.

In this example, a parameter, PAppRampRef, will be defined that will be used to obtain the speed reference for the drive.

Another read-only parameter, vCount, will be used to show the counter value.

The unit of measurement of the new parameter will be “% of Full Scale Speed” and conversion from % to the internal in count values of the drive will also be performed.

1) Select the “Params” card of the “Project window”.

2) Open the “Test00 parameters” structure.

3) Double-click on the “Parameters” folder. The parameter editor is opened.

4) Add a new parameter, selecting the option of the “Parameters” + “Insert” menu.

5) The dialogue box will be displayed for entry of the parameter. Select “32 bit” and press OK. An empty parameter record will appear in the editor.

6) Click on the “Name” field and enter “AppRampRef” which will be the name of the new parameter.

7) Set the field of the menu to “Application”, this is a pre-defined menu deriving from installation of the ADV200 support file.

8) Other menus can be added inside the “Menus” folder of the “Params” structure.
9) Click on the “Type par” field and select “Float” as type of data. This means that the parameter exchanged with E@syDrive/GF_eXpress or shown on the keypad will be of the Floating point type (at 32 bit).

10) Set the “Type targ” field (long). This means that the parameter will be represented internally as a 32-bit integer.

11) Compile the “Min” and “Max” fields with −100 and 100 respectively. This prevents setting of incorrect speed references. The values entered will be limited within these two values +/-100% of the Full scale speed parameter.

12) Inside the drive, the speed scaling factor is such that 4000H * 2^16 = Full Scale Speed. In other words, 1024*16384 = 1073741824 as sysRampRef produces a speed reference equal to the value of the parameters of drive Full Scale Speed.

   - At this point, set the “Scale” field with the appropriate conversion factor from “% of FS” to internal units of the reference speed. Scale must be such that 100*scale = 1073741824.

   - Scale = 1073741824/100= 1073741.8. This is the value to be set in the “Scale” field.

13) Enter “%FSpd” for the “Unit” field and “Speed reference as % of the full scale speed” for “Description”. The record of the parameter should appear as shown below (divided into two parts).

<table>
<thead>
<tr>
<th>IPA</th>
<th>Addr</th>
<th>Menu</th>
<th>Name</th>
<th>Attr...</th>
<th>Type Par</th>
<th>Type Targ</th>
<th>Def. value</th>
<th>Min</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>11000</td>
<td>%MD1.0</td>
<td>APPLICATION</td>
<td>AppRampRef</td>
<td>UE</td>
<td>Float</td>
<td>Long</td>
<td>0</td>
<td>-100</td>
<td>100</td>
</tr>
<tr>
<td>3700</td>
<td>Pad 1</td>
<td>...</td>
<td>Long</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3702</td>
<td>Pad 2</td>
<td>...</td>
<td>Long</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Scale</th>
<th>Offs</th>
<th>Unit</th>
<th>Short Description</th>
<th>Description</th>
<th>Note</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>107772</td>
<td>0</td>
<td>%FSpd</td>
<td>AppRampRef</td>
<td>Speed reference in % della full scale</td>
<td></td>
<td>$d</td>
</tr>
</tbody>
</table>

14) Similarly, define the Count read-only parameter.

   - As opposed to the first, I will use the “Read only parameters” folder. As this is a counter, I will not insert any scale. At the end, I will obtain:

<table>
<thead>
<tr>
<th>IPA</th>
<th>Addr</th>
<th>Menu</th>
<th>Name</th>
<th>Attr...</th>
<th>Type Par</th>
<th>Type ...</th>
<th>Scale</th>
<th>Unit</th>
<th>Short Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>12000</td>
<td>%MEM.0</td>
<td>APPLICATION</td>
<td>Count</td>
<td>...</td>
<td>Int</td>
<td>Int</td>
<td>0</td>
<td>Count</td>
<td></td>
</tr>
<tr>
<td>4454</td>
<td></td>
<td>Bit0 decom...</td>
<td>Word</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4455</td>
<td></td>
<td>Bit1 decom...</td>
<td>Word</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Short Description</th>
<th>Description</th>
<th>Note</th>
<th>Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contatore</td>
<td>valore attuale contatore</td>
<td></td>
<td>$u</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>$u</td>
</tr>
</tbody>
</table>
7.11 USE OF THE PARAMETERS IN THE APPLICATIONS

After creating the new parameter, it can be used in this simple project. There will also be a sort of data exchange between the external world (keypad, configurators, supervisors, etc.) and the application.

For each parameter, MdPlc automatically generates a global project variable that can be used directly by the blocks of code (as regards PROGRAMS) or through external declarations (as regards FUNCTION BLOCKS).

The procedure below describes how to use the new parameter inside the “Slow” program and its use as speed reference.

1) Save the document of the parameter with “File” + “Save”. The “Addr” field is automatically compiled with a suitable address in the memory of the drive.

2) In the “Project” window, select the “Project” sheet again.

3) Double-click on “Project parameters” of the “Project” window. Entering DINT, the folder will open showing the new variables called “pAppRampRef” and vCount. These variables are associated to the AppRampRef and Count parameters.

4) Open the “Slow” program again in the code editor, for example double-clicking on its icon in the “Project” window.

5) The speed reference for the drive consists of the “sysRampRef” variable. It is possible to find this in “Target variables”. Move it or enter its name on the next available line of the “Slow” program.

6) Complete the instructions assigning the value of the “pAppRampRef” parameter to “sysRampRef”. The complete instruction will be “sysRampRef := pAppRampRef;”.

   - I assign the counter value to vCount.
      - The editor window should appear as shown in the figure below.
- The `vCount := counter;` instruction assigns the counter value to a read-only parameter (the v prefix identifies this as read-only). These parameters can be seen both on the keypad and via GF_eXpress.

- `pAppRampRef` is a read/write parameter (the p prefix identifies this as R/W parameter). These parameters can be seen both on the keypad and via GF_eXpress. They are also saved in the flash memory when saving parameters.

7) Compile the code and download this again. The drive will be reset automatically.

8) Start the GF_eXpress configurator, open the “Test00.par” file again and make the connection to the drive. Click to activate the “Application” menu. The list of parameters will now also show the parameters just created.

9) The keypad of the drive will also display this new parameter in the related menu.
10) Set the value of *pAppRampRef* to 0.

11) Go to MdPlc and move “sysRampRef” and “pAppRampRef” into the watch window (see paragraph 7.8 point 3). Their value will be equal to 0.

12) Change the value of the *pAppRampRef* parameter to 100 %. The *sysRampRef* will go to its maximum value in count.

13) Using digital input 1, it is possible to switch between the two values.
8 OTHER PROGRAMMING OPTIONS

The previous paragraphs have described the first steps in MdPlc programming. MdPlc applications are obviously much more complex than those described in the example.

To help programmers develop applications of any level of complexity, MdPlc features various capabilities able to facilitate this task, making it possible to carry it out more quickly and simply.

Other elements relating to development of an MdPlc application not covered in this quick guide but which are however important enough to be taken into account are listed below.

- **The languages**
  The IEC languages, in addition to the ST language, are useful for catering to the requirements of the SFC application; in particular, it is suitable for organizing the code in the execution phases and for constructing codes that are easy to read and understand.
  It is important to remember that, in a single application, all the languages can be used together and each POU can be written using any IEC language.

- **Functions and functional blocks**
  The real importance of IEC programming stems from the modularity of the code. The functions and functional blocks make it possible to create a modular, portable code. MdPlc fully supports the functions and functional blocks, permitting nesting of the blocks at any level in all five IEC languages.

- **The libraries**
  The libraries are the supporting blocks of the development of an application. The MdPlc setup installs various libraries referring to the basic blocks (flip-flop, edge detectors, timers etc.), CAN communications, control (PID, filters, etc.), position control and so on.
  Also, MdPlc provides programmers with various possibilities of creating their own library and of importing and exporting the blocks between the projects and libraries.

- **Real-time debugging**
  Real-time debugging is one of the most important characteristics of MdPlc which guarantees a fast, precise debugging procedure.
  With the triggers and the graphic trace option, it is possible to identify particular conditions of execution and to track the behavior of the variables during execution of the code, without stopping the target or modifying execution times.

- **On-line debug**
  The on-line debug procedure is available for SFC and LD code editors. It provides an “active” representation of the graphic elements to facilitate understanding of the flow of the program.
• **Parameters**
  Parameter support does not only include the few characteristics presented in this guide but also enumerative events, menus, conversion expressions, R/W events, etc..

• **Editing capabilities**
  MdPlc also offers a vast range of code maintenance and editing possibilities: searching of the project, printing of the project, macros, editors of the bookmarks, object browsers and others.
9 SOFTSCOPE

Real time Oscilloscope, it can be installed from CD or setup file.

SoftScope is a software designed for the sampling of the drives and MDPLC variables and displayed in real time (like on oscilloscope).

For further information about this tools, refer to Softscope manual.