



# **CANmaster PC-Tool User Guide**

**Developer Edition**

# Table of Contents

## 1. Introduction

<b>Welcome to CANmaster PC-Tool</b> .....	<b>1</b>
Scope of this manual and other documentation .....	1
Program Versions .....	2
<b>Installing CANmaster PC-Tool</b> .....	<b>2</b>
Hardware lock .....	2
System requirements .....	2
Starting CANmaster PC-Tool .....	2
<b>Main steps in a development project</b> .....	<b>3</b>
System specification .....	3
Program development .....	3
Testing and verification .....	3

## 2. CANmaster PC-Tool - Basics

<b>Main window</b> .....	<b>5</b>
Menus .....	5
Toolbars .....	6
<b>Project Browser</b> .....	<b>10</b>
Files .....	10
Parameters .....	10
Units .....	11
Symbols .....	11
<b>Programming Window</b> .....	<b>11</b>
<b>Message Window</b> .....	<b>12</b>

## 3. Working on a project

<b>General</b> .....	<b>13</b>
<b>Creating a new project</b> .....	<b>13</b>
<b>Open a saved project</b> .....	<b>15</b>
<b>Files and resources</b> .....	<b>16</b>
Control units .....	16
Source code files .....	17
Project browser file manager context menu .....	18
Menu files for the terminal .....	18
<b>Resulting program files</b> .....	<b>19</b>
<b>Version management</b> .....	<b>20</b>
Product versions .....	20
Version number .....	20
Version dependency .....	21
Revision control of source code files .....	22
Project delivery .....	22

## 4. Design of application programs

<b>General</b> .....	<b>23</b>
<b>Defining inputs and outputs</b> .....	<b>23</b>
Symbol names .....	23
Configuring I/O .....	23
<b>Graphical programming</b> .....	<b>24</b>
Programming window .....	25
Function blocks .....	25

Grouping .....	27
Viewing connections .....	27
<b>Order of execution .....</b>	<b>27</b>
Local order of execution .....	28
Global order of execution .....	28
<b>Configuring parameters .....</b>	<b>28</b>
<b>Data types .....</b>	<b>28</b>
<b>Text-based programming .....</b>	<b>29</b>
<b>Compiling program files .....</b>	<b>29</b>
<b>5. CAN Configuration</b>	
<b>Nodes .....</b>	<b>31</b>
<b>Messages .....</b>	<b>32</b>
Name .....	32
PGN .....	32
DLC .....	32
Transceive Method .....	33
Condition .....	33
Comment .....	34
Message content .....	34
<b>Browsing standard messages .....</b>	<b>35</b>
<b>6. Terminal Design</b>	
<b>General .....</b>	<b>37</b>
<b>Main window in Terminal Design .....</b>	<b>37</b>
Menu structure and navigation .....	38
Buttons for navigating between the menus .....	39
Global settings .....	40
Screen visuals - general .....	41
Terminal objects .....	42
<b>Design of screen visuals .....</b>	<b>42</b>
Passive terminal objects .....	43
Active terminal objects .....	43
Icons and fonts .....	45
<b>Multiple languages .....</b>	<b>46</b>
<b>Terminal script .....</b>	<b>46</b>
Terminal scripts in general .....	46
Enhanced terminal functions .....	47
Terminal script and menus .....	47
Configuration parameters .....	47
<b>Compiling terminal programs .....</b>	<b>48</b>
<b>7. Display programming</b>	
<b>Files .....</b>	<b>49</b>
<b>Interface Overview .....</b>	<b>49</b>
<b>Designing an application .....</b>	<b>50</b>
<b>Configuration of functionality .....</b>	<b>50</b>
Keyboard input .....	50
<b>Scripting .....</b>	<b>50</b>
Application scripts .....	50
<b>8. Parameters</b>	
<b>General .....</b>	<b>53</b>

<b>Unit specific parameters</b> .....	<b>54</b>
Joystick parameters .....	56
<b>Parameter editing</b> .....	<b>57</b>
<b>Parameter Tree</b> .....	<b>59</b>
<b>Creating Parameter Editors</b> .....	<b>60</b>
<b>Compare and merge parameters</b> .....	<b>63</b>
<b>9. Compiling projects</b>	
<b>Compiling application program</b> .....	<b>67</b>
<b>Correcting compilation errors</b> .....	<b>68</b>
Errors in source code files .....	68
Utilized memory .....	68
<b>Compiler settings</b> .....	<b>68</b>
<b>Downloadable files</b> .....	<b>69</b>
<b>10. Testing projects</b>	
<b>Simulate projects</b> .....	<b>71</b>
Simulation of control logic .....	71
Open the simulator function .....	72
Simulation methods .....	73
Displaying variable values .....	74
Simulator scripts in general .....	76
Monitoring CAN-bus signals .....	77
Simulation of terminal program .....	78
<b>Monitoring signals</b> .....	<b>79</b>
Text based monitor .....	79
Graphical monitor .....	79
<b>11. Downloading software</b>	
<b>Quick reference</b> .....	<b>83</b>
Downloading an application .....	83
Uploading parameters .....	84
<b>Subdivision of files</b> .....	<b>85</b>
<b>Downloading various items of software</b> .....	<b>86</b>
Downloading application program and parameters .....	86
Downloading basic software .....	86
Download of application program via Master/Terminal .....	86
Download of program via bootstrap .....	87
<b>12. Error reports</b>	
<b>Uploading error logs</b> .....	<b>91</b>
<b>Error-log window</b> .....	<b>92</b>
Description of the information in the error-log window .....	92
<b>Saving error logs</b> .....	<b>93</b>
<b>Internal error messages</b> .....	<b>93</b>
Emergency .....	94
Voltage & Temp .....	95
CAN-bus errors .....	96
Joystick unit .....	97
Error .....	98

## Index



# Introduction

---

## Welcome to CANmaster PC-Tool

CANmaster PC-Tool is a complete development tool for designing and maintaining control programs for the CANmaster® control system.

The CANmaster PC-Tool includes all the parts required to create a working control system for your machine, including features for graphic design of application and terminal programs, source code editors, compiling programs, program simulation, logging, downloading software, version management and program documentation. The program also contains the features required for advanced service and maintenance of the system for the machine in service. It combines all of this in an easy to use graphical environment.

You can use this versatile program to create everything from simple control systems with a few input and output signals to complex machine systems with integrated control and monitoring of, for example, diesel engines, hydrostatic transmissions and complete working hydraulics systems.

The CANmaster PC-Tool allows development of application programs in both graphical and text based source code, enabling you to choose to develop your programs in the way that feels most secure and natural. The program is constructed in the graphical development environment using graphical blocks that simplify the programming process substantially, providing a better overview of the program features and reducing the need of detailed prior knowledge of programming microprocessor based control systems.

## Scope of this manual and other documentation

The CANmaster PC-Tool User Manual - Developer, contains information on all parts of the CANmaster PC-Tool. The manual is intended for use by program developers as a reference in their day to day work with the CANmaster control system and programs.

The manual describes the various parts of the development tool in detail and how these can be utilised. However, the function blocks that are available in the CANmaster PC-Tool for rationalising the process of producing application programs are not described in detail in this manual which only includes the main features and working methods to ensure they are utilised in the programming process in an efficient way. A detailed function description of the function blocks is available in the program's built-in help facility and in the separate reference manual entitled 'CANmaster PC-Tool Function Library Reference'.

Parts of the content in this manual have been issued in versions that are intended for service and production staff with information concentrated on the parts of the program that relate to the relevant authorisation level.

Before using the manual and starting the program, it is useful to have read the manual 'CANmaster Technical Data & Installation' which contain basic technical data, connection principles and I/O lists for control units in the CANmaster control system.

## Program Versions

The CANmaster PC-Tool is available in a number of versions with different levels of authorisation. The program version for service staff, for example, has no option for changing or compiling the control system's application programs, but has the option of adjusting parameters, downloading updated software, reading fault logs, logging operational data, calibrating sensors etc. All parts of the program are available for system developers. The various program versions are all contained in the same program concept where authorisation to the different parts of the program is fully controlled by a hardware lock that is delivered with the program.

## Installing CANmaster PC-Tool

The CANmaster PC-Tool must be installed on your computer's hard disk and cannot be run from the CD. Run the installation program and follow the on-screen instructions. A shortcut to the program is automatically created on the computer desktop and in 'Program' in the Windows Start menu.

### Hardware lock

Starting the program requires a valid hardware lock that checks for a valid licence and the part of the program to be run. Your hardware lock with relevant authorisation was included in the delivery of the program disk and this manual, and is to be connected to the computer's USB port.

### System requirements


The table below gives the minimum requirements for computer hardware and software to enable you to use the CANmaster PC-Tool.

Table 1.1. System requirements.

System requirements user computer for CANmaster PC-Tool	
Computer type	PC compatible
Operating system	Microsoft Windows® 98/2000/XP (The CANmasterTool setup program requires that Windows Installer 2.0 is installed on the computer. Later operating systems have it included but if Widows 98 is used, it might have to be installed)
RAM-memory	Min. 128 MB for Windows® 98. More than 256 MB for later versions of the Microsoft Windows® operating system
Processor	800 MHz
Hard disk space	25 MB (software for CANmaster PC-Tool)
Communication ports	RS232 and USB
Screen resolution	Program development: recommended min. 1280 x 1024 pixels. Service work: recommended min. 800 x 600 pixels

### Starting CANmaster PC-Tool

To start CANmaster PC-Tool, do one of the following:

- Double-click on the CANmaster PC-Tool icon on the computer desktop.
- Select Start - Program - Hydratronics - CANmasterTool.
- Double-click on a CANmasterTool project file  from Windows explorer.

The program starts and displays the main window with grey tool bar at the top of the screen.

## Main steps in a development project

Work on a system development project is normally divided into three main steps; Basic system specification, Program development and the final step, Testing and verification. All of these parts of the project require the work to be structured in a way that ensures the shortest time possible for producing a control system that is capable of handling the set goals for machine functionality and performance.

### System specification

Before starting to work on your development project in CANmaster PC-Tool, you should have access to a separate system specification for the machine with a detailed description of the machine functions that the control system must cater for, and how these parts are mainly to be controlled.

As the selection of control units in most cases is dependent on the number of power outputs required for controlling the various types of valves, on/off controlled or PWM controlled valves and the location of the valves on the machine, these parts must be clearly defined at an early stage of the project.

The Master unit must always be included in a CANmaster control system, which is why the selection of control units applies to the selection of the other control units that may be required; Expansion, Crane, Joystick H5-S50 and Terminal T2. The expansion unit is used primarily to add more analogue and digital inputs, while the Crane unit adds more PWM outputs. Each Joystick H5-S50 has the capacity to handle 10 digital inputs, and also has 2 analogue inputs for connecting separate joysticks or potentiometers.

The CANmaster control system has a lot of excess capacity on the digital and analogue inputs, which for the most part can also be configured as either digital inputs or analogue inputs. The system specification does not therefore need to be fully complete with respect to these parts from start, but can be specified more accurately later on.

The final system specification with detailed function descriptions for each machine function, associated valves, operating mechanisms and sensors make up the basis for both application programs and terminal management and an important document that should be kept updated as the work progresses and new ideas for controlling and monitoring the machine system emerge. The function descriptions that may initially consist solely of descriptions in text format, should also be supplemented with a graphical description of output signals as a function of input signals whenever possible to ensure a clear picture of how the various functions are to be controlled and what main conditions apply. The document serves as a reference for discussions on function principles and program conditions without the need to go into detail on how this is to be implemented from a purely programming perspective.

### Program development

When the basic system specification is ready, it is then time to start to work on the project in CANmaster PC-Tool. Figure 1.1, "Working procedure for producing a project with CANmaster PC-Tool." provides a general view of the procedure for producing a project using CANmaster PC-Tool.

### Testing and verification

The CANmaster PC-Tool has the option of testing the control program thoroughly before downloading to the control units using the built-in control program simulator. The simulator works with the same compiled program files as those used in the control units. The control program therefore works in exactly the same way in the simulator as if it had been downloaded into the control units in the machine. The possibility to create test script files allows the development of testing methods that are very close to realistic operating conditions. Consequently, when you download your control program to the machine for testing and verifying, a lot of the basic testing has already been done and you can concentrate on refining the various features of the program.



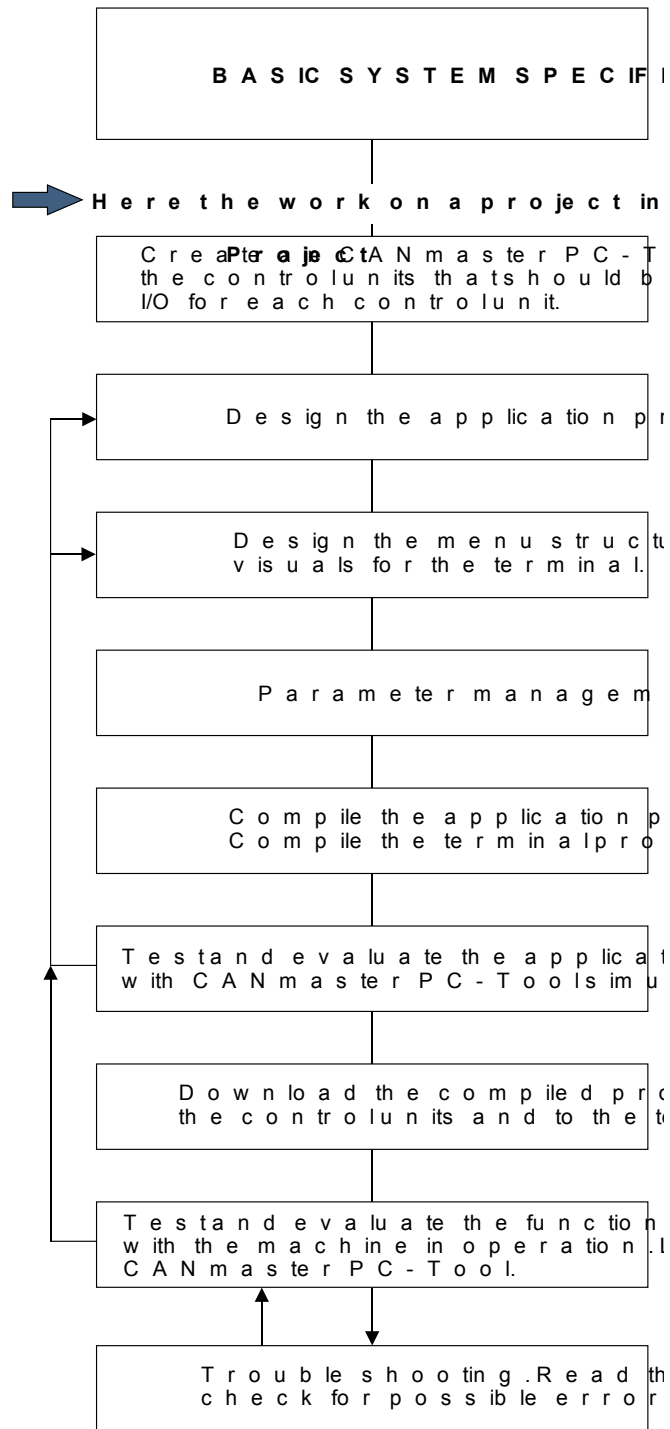


Figure 1.1. Working procedure for producing a project with CANmaster PC-Tool.

# CANmaster PC-Tool - Basics

---

CANmaster PC-Tool is based on a number of windows that you position as you prefer them on your screen. The program is divided into three main categories; Main window, Tool window and Work spaces. In addition to the basic windows, you will also use different settings and configuration windows that are only viewed while you carry out a particular task. These windows are displayed as dialogue boxes.

This chapter gives an introduction to the development environment's full capacity and a general description of the various parts of the program that are available in the form of toolbars, menus and as direct commands to enable you to get working quickly and efficiently with CANmaster PC-Tool.

## Main window

The program's main window is displayed at the top of the screen when the program starts. You can view the main menus of the program at the top of this window with toolbars under these. The menus alter in appearance a little depending on the work space that is currently active.

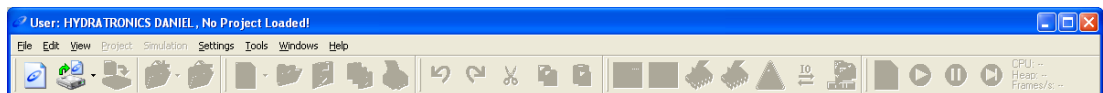


Figure 2.1. Main window.

## Menus

The menus are logically divided up per function, and each sub-menu can have groups of functions in order to provide a quick overview of the possibilities the program offers. When a certain menu option is not available it is greyed out while available commands appear in black.

### File

All the functions associated with a project are available under the File menu. These include functions for opening and closing projects, as well as functions for opening and saving separate files and printing them.

### Edit

The Edit menu includes the options for handling changes. Here you have the option of undoing, repeating a command and handling the clipboard by copying, cutting and pasting information between different work spaces or other programs.

### View

You can view and hide the various tool windows that CANmaster PC-Tool consists of by selecting or deselecting them in the View menu. A marker by an option means that the current selection is active.

The windows you can view and hide from this menu are the Project browser, Message window and the Graphic tool palette.

## Project

The Project menu contains functions for handling the different types of file included in the project, as well as handling the properties of the project. Here you have the commands for adding and removing source code files in the project, and setting the control units the project is to include.

Here you will also find the functionality to export and change the name of the project as well as handling versions of the project. The function for Build & Compile project is also available under this menu.

## Simulation

All functions to control a simulation are here.

## Settings

The general options affecting the entire program are available under the Settings menu, including the option of setting the communication port to be used when communicating with the control units. There are also different help tools here for configuring inputs and outputs, adjusting parameters and handling service functions such as reading of error logs or downloading software for the control units.

## Tools

Special tools. The Communication terminal allows you to directly communicate with the target system. The Remote connection brings up a dialog window where you can connect to a remote target system through a modem.

## Windows

Different help functions for handling, moving around and viewing the various windows that are open, are available in the Windows menu. Here you will find the various options for minimising all windows or placing them in tile format.

## Help

The Help menu gives access to the program's built-in help function. Here you will also find the option for reviewing the version of CANmaster PC-Tool and its components that you are currently using as well as the licence that is active.

## Toolbars

A number of toolbars are available under the menus in the main window. These buttons carry out the same function as the most frequently used menu options and are there to make the program more versatile and easier to use. The various toolbars and the functions the buttons have are described below.

The program window displays a brief help text describing the function of each button when you hold the cursor still on a button. You can move the various toolbars around yourself so that they are positioned as per your wishes by gripping the vertical handle that is located far left in each toolbar. If you pull a toolbar outside of the main window, you create an independent window that you can position wherever you like on screen. This position can be reset by gripping the blue title bar and pulling the window back to its original place.

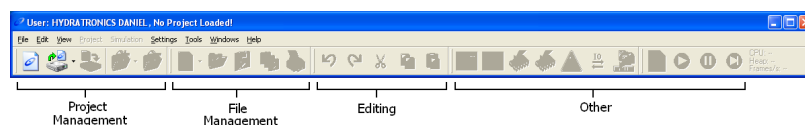


Figure 2.2. Available toolbars in main window.

## Project related functions

The toolbar that belongs to the project related functions is located far left:

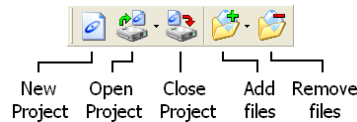


Figure 2.3. Project management toolbar.

The buttons have the following function:

New project	Use this button to create a new project. If a project is open when you press the button, the current project is saved and the guide to create a new project then starts. The corresponding menu option is File -> New Project ...
Open project	Use this button to open a previously saved project. Pressing this button opens a file selector dialogue where you can navigate to the project file you want to open. If another project is open, it is saved and closed before the new project is opened. The corresponding menu option is File -> Open Project ...
Close project	Use this button to close the current project. If the project contains changes, you will be asked to save these before continuing. Depending on your selection; (1) the project is saved and then closed, (2) the project is closed without saving the changes, or (3) the project remains open. The corresponding menu option is File -> Close Project...
Add to/Remove from	To add files or remove files from an open project, you can click on this button and then select the type of file you want to add/remove. A file selector dialogue then opens where you are given the option of selecting the files you want to add/remove to the project.

### Tip

Add/Remove file: You can also do this by right-clicking on a file in the project browser and selecting Add to Project or Remove from Project in the context menu that appears.

## Edit functions

The toolbar for editing contains the functions most frequently used for changes in your daily work:

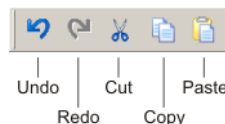


Figure 2.4. Edit toolbar.

The function of the buttons is:

Undo	Undo the latest change. The same function can be reached via the menu option Edit -> Undo ( <b>CTRL+Z</b> ).
Redo	Press this button if you have undone an action, but want the action done again. The same function can be reached via the menu option Edit -> Redo.

- Cut** Press this button to cut the selected text or function block. The same function is available via the menu option Edit -> Cut (**CTRL+X**).
- Copy** Press this button to copy the selected text or function block. The same function is available via the menu option Edit -> Copy (**CTRL+C**).
- Paste** Click this button to paste previously copied or cut material. The same function is available via the menu option Edit -> Paste (**CTRL+V**).

## File related functions

In order to handle the various tasks related to files, there is a toolbar with shortcuts to the most frequently used functions.

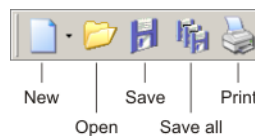


Figure 2.5. File toolbar.

The function of the different buttons is:

- New file** To create a new file, you can either choose to press this button or go via the menu and select File -> New. In both cases another menu opens with more options for creating different types of file. The different file types that can be created are:
- Text based source code
  - Graphic design
  - Menu definition
  - Script file
- Open file** Pressing this button opens a file that already exists. After you have pressed this button a file selector dialogue is displayed that allows you to navigate to the required file. The same function is available via the menu option File -> Open .
- Save changed file** The button for saving a file is only available if the active window contains some changes. If the content has changed, the file is saved to the hard disk. The same function is available via the menu option File -> Save (**CTRL+S**).
- Save all changed files** This button is available if any one of all the files that are open contains a change. Pressing this button saves all the windows that have been changed to their relevant files on the hard disk. The same function is available via the menu option File -> Save All (**CTRL+SHIFT+S**).
- Print** Use this button to print the contents of the current window. Pressing the button once opens a dialogue box with printer settings where you can select printer and paper format etc. The same function is available via the menu option File -> Print (**CTRL+P**).

## Other shortcuts

Other functions that are frequently used have been grouped in individual toolbars. These functions are primarily aimed at testing and service but the toolbar does include other functions.

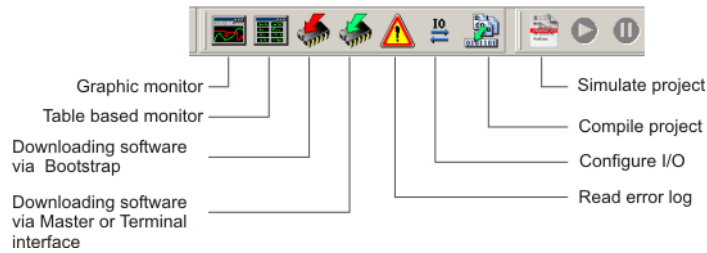


Figure 2.6. Other commonly used functions toolbar.

Function of the buttons:

Graphic monitor	This button opens a graphic monitor. The graphic monitor connects to a control unit unless the program's simulator is running whereby the monitor works with the simulator instead. For more information on testing and measuring, see the section called "Graphical monitor".
Table based monitor	This button opens a window that can be used to get an overview of all signal values in the system. For more information on testing and measuring, see the section called "Text based monitor".
Downloading via Bootstrap	Use this button to download software to a control unit using bootstrap (direct connection of the service computer to the RS232 port on a control unit). For more information on downloading of control programs and other types of files for the control units, see the section called "Download of program via bootstrap".
Downloading software via Master or Terminal interface	Use this button to download application programs and parameter files for the control units via the RS232 port on the Master Unit or the Terminal. For more information on downloading of programs and other types of files for the control units, see the section called "Download of application program via Master/Terminal".
Simulate project	Activate this button to open the simulator function with the current project. All programming windows are set in simulator mode and cannot be changed when the simulator is running. For more information on testing and simulation, see the section called "Simulate projects".
Read error log	Press this button to read an error log from a connected control unit. For more information on error reporting, see Chapter 12, <i>Error reports</i> .
Configure I/O	Use this button to configure the input and output signals that are to be used for the control units and the corresponding symbol names. For more information on how you define the input and output signals, see the section called "Defining inputs and outputs".
Compile project	Press this button to compile the entire control program into executable binary files that can be downloaded to the control units. For more information, see Chapter 9, <i>Compiling projects</i> .

### Simulator functions

This toolbar is only shown when the simulator in CANmaster PC-Tool is activated.

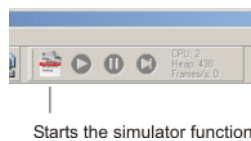


Figure 2.7. Simulator functions toolbar

The function of the buttons and the values displayed for CPU, Heap and Frames/s are described in detail in the section called "Simulate projects".

## Project Browser

When CANmaster PC-Tool is started a separate project browser is displayed in addition to the main window.

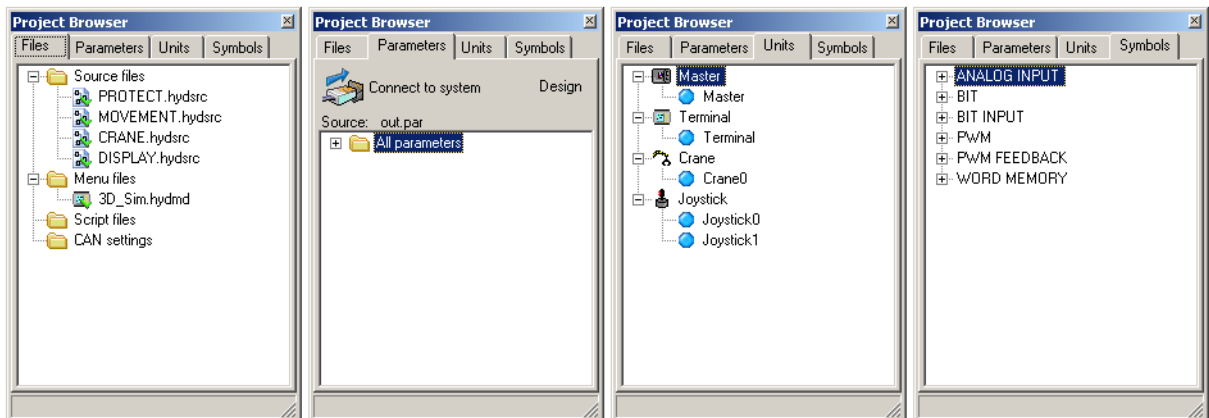


Figure 2.8. The different tabs of the project browser.

The project browser contains all information about the current project, the resources included in the project, the control units included and their versions as well as parameters and symbols. The information is divided into a number of tabs as shown in the figure above which are described below.

### Files

The Files tab has a list of all the files included in the project. These files are in turn divided further into a range of categories depending on the type of file. There are four categories of file in a project:

- |                   |  |
|-------------------|--|
| Source code files | Both text and graphic based source code files are grouped together. You can see the type of source code file from the icon in front of the file name.            |
| Menu files        | Menu files contain definitions of how the terminal's menu system is designed.  |
| Script files      | A project may include different types of command files in the form of script files used for testing, among other things. These files are grouped under this tab. |

All files in the project directory and that belong to one of these categories are in the list of files. If the file is in the project directory but is not included in a specific project, the file is greyed out.

### Parameters

The Parameters tab displays all the parameters that can be configured in the system. They are displayed in a tree view where each parameter can be found under a drop down header. The header comes from the name selected for the current function and all parameters that are associated with the function are displayed with name and value.

The designer can also create a customized parameter view to be used by service personnel. In this tree view, parameters can be arranged in a way suitable for the application.

Handling parameters makes up the lion's share of the development work and subsequent running adjustments for the machine application. For more information on how the CANmaster PC-Tool can simplify the handling of parameters, see Chapter 8, *Parameters*.

## Units

The control units included in the project are displayed in a tree view under this tab. Each type of unit is grouped individually. The selected version of the base software used in the units and the number of units of each type are displayed for each type of unit.

## Symbols

Normally there are quite a lot of symbols to keep track of in a project. A 'symbol' can be an internal variable an input or output. The symbols can be viewed under the Symbols tab, grouped according to the type of information the symbol refers to.

## Programming Window

The bulk of the development work is carried out in some kind of editor. A normal text editor with enhanced functionality to mark key words etc. is used for the text based programming. This is nothing new if you have previously worked with different programming languages. If you are not used to writing programs directly as textbased programming code you will not need to learn this now. The CANmaster PC-Tool has support for designing control programs using graphical aids. This is done in a graphic editor where you can drag function blocks from a graphical browser and drop them into the design area. Using the cursor you can link the I/Os for the different function blocks with other function blocks by drawing lines similar to designing a connection diagram for an electrical system to achieve the required effect.

All editors are displayed in the same window, under separate tabs.

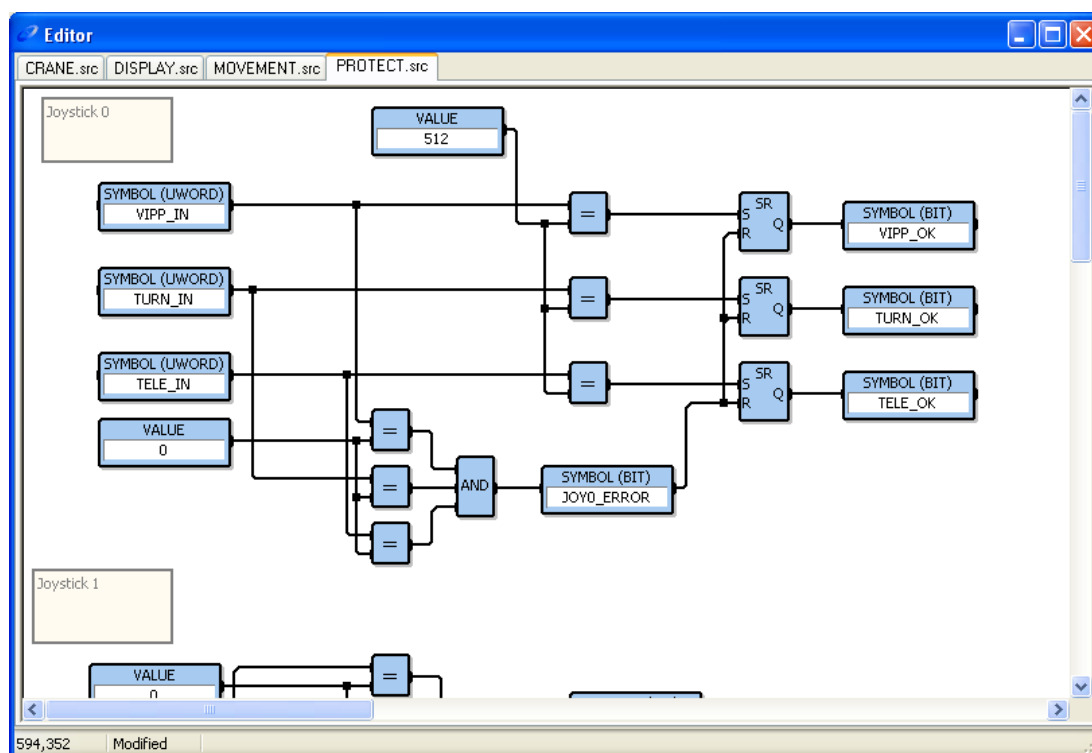
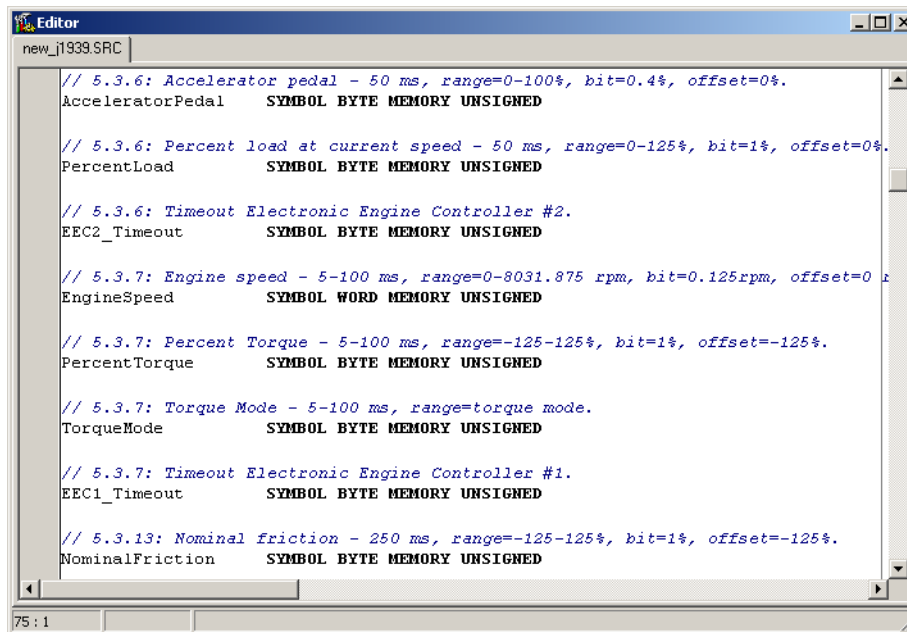


Figure 2.9. Graphical code editor. The graphical browser is displayed in a separate window with drag & drop functionality for the different function blocks.



The image shows a window titled 'Editor' with a file name 'new\_11939.SRC'. The window contains several lines of code defining symbols for CAN messages. Each line consists of a comment and a symbol definition. The symbols are: AcceleratorPedal (BYTE MEMORY UNSIGNED), PercentLoad (BYTE MEMORY UNSIGNED), EEC2\_Timeout (BYTE MEMORY UNSIGNED), EngineSpeed (WORD MEMORY UNSIGNED), PercentTorque (BYTE MEMORY UNSIGNED), TorqueMode (BYTE MEMORY UNSIGNED), EEC1\_Timeout (BYTE MEMORY UNSIGNED), and NominalFriction (BYTE MEMORY UNSIGNED). The status bar at the bottom left shows '75 : 1'.

```
new_11939.SRC
// 5.3.6: Accelerator pedal - 50 ms, range=0-100%, bit=0.4%, offset=0%.
AcceleratorPedal    SYMBOL BYTE MEMORY UNSIGNED

// 5.3.6: Percent load at current speed - 50 ms, range=0-125%, bit=1%, offset=0%.
PercentLoad        SYMBOL BYTE MEMORY UNSIGNED

// 5.3.6: Timeout Electronic Engine Controller #2.
EEC2_Timeout       SYMBOL BYTE MEMORY UNSIGNED

// 5.3.7: Engine speed - 5-100 ms, range=0-8031.875 rpm, bit=0.125rpm, offset=0
EngineSpeed        SYMBOL WORD MEMORY UNSIGNED

// 5.3.7: Percent Torque - 5-100 ms, range=-125-125%, bit=1%, offset=-125%.
PercentTorque      SYMBOL BYTE MEMORY UNSIGNED

// 5.3.7: Torque Mode - 5-100 ms, range=torque mode.
TorqueMode         SYMBOL BYTE MEMORY UNSIGNED

// 5.3.7: Timeout Electronic Engine Controller #1.
EEC1_Timeout       SYMBOL BYTE MEMORY UNSIGNED

// 5.3.13: Nominal friction - 250 ms, range=-125-125%, bit=1%, offset=-125%.
NominalFriction    SYMBOL BYTE MEMORY UNSIGNED

75 : 1
```

Figure 2.10. Textual code editor.

## Message Window

Another type of standard window in CANmaster PC-Tool is the message window, where you can view messages about the project. The most common use of this is during compilation of the control program where the PC-tool program displays error messages if an error has occurred. Clicking on a message and pressing function key **F1** gives access to more information on the message. You can also double-click the message row and an appropriate editor shows the location of the error in the source code file. However, if everything works as it should during the compilation, messages on the size and memory utilisation of the control program are displayed.

# Working on a project

---

## General

The work on developing an application program for a machine always starts by creating a Project. A project in CANmaster PC-Tool is divided into a number of separate files to get a better overview of the control program and greater flexibility in the design work. The CANmaster PC-Tool.

An application program in CANmaster PC-Tool is divided into a number of separate program files at different levels so as to create a better overview of the complete control program and provide greater flexibility when programming. CANmaster PC-Tool groups and keeps track of all the component files and their internal dependency where both source code files and general information on the project is collected in a common map structure: a project.

Programming in CANmaster PC-Tool is chiefly graphical where separate program files known as source code files are brought together in a Project that is then compiled to enable downloading to the control units.

In addition to pure source code files that make up the basis of the application program, the project also includes information on program version and project name, the control units included in the project and how inputs and outputs are configured for the control units included in the project.

This chapter describes how work on the project and file management works in CANmaster PC-Tool, and outlines the support that is built-in to the program to handle the different versions of application programs and updates.

Note that you can only work on one project at a time and each project must be located in its own master folder.

## Creating a new project

As all work on CANmaster PC-Tool revolves around a project, you must start by creating a Project. A guide is built in to help you through the process where you enter the working directory for saving the project files, the name of the project, the version number and the control units to be included in the project.

To create a new project:

1. Select File -> New -> Project...
2. Select the working directory the project is to be saved under. You can browse to a suitable location by clicking the button to the right of the data entry field.
3. Create a new project folder under the selected working directory. You can create new folders in the dialogue box that you view by clicking Create new folder and entering the name of the new project folder.

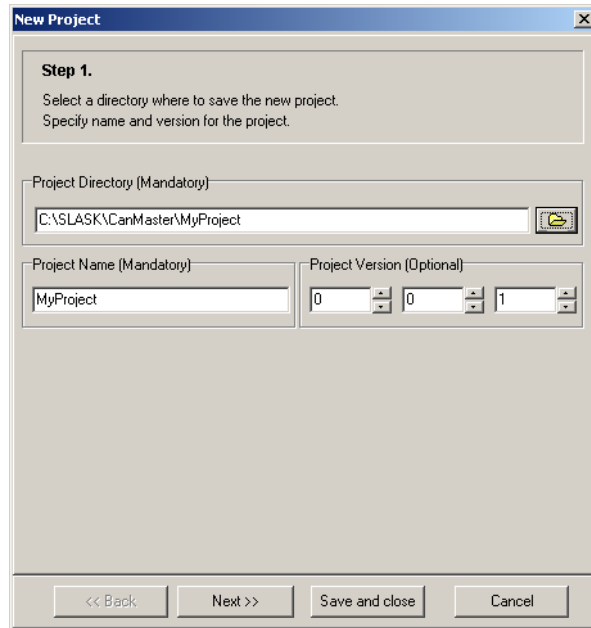


Figure 3.1. First step in creating a project. Enter the folder here for saving the project on the hard disk along with the project version number.

4. Select project name in the box under Project Name. The project name should be the same as the name you used for the project folder in the previous step.
5. Select the project version number using the scrollbars for the version number or enter the numbers directly.
6. Select Next>>.
7. Here you can select the control units to be included in the project. You can also specify the version of hardware and base software (Boot- and Middleware) for the selected control units.

The master unit is selected by default as a CANmaster project normally uses at least a master unit. The only time a project should be without a master unit is when a you are developing an application for an extended slave unit.

## Note

All settings done in this view can also be done at a later stage if you prefer, see the section called “Control units”.

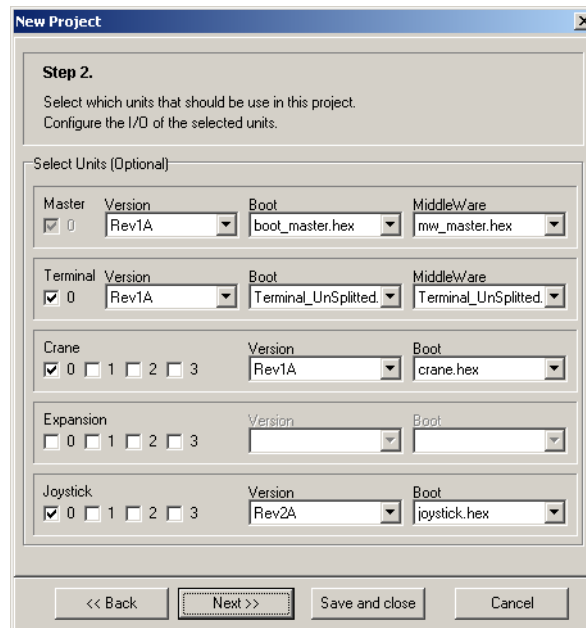


Figure 3.2. You enter the control units you will be using in the project here. You should also select the version of both hardware and base software for the selected units.

8. Select Save and close.

The Project has now been created and you can start work on the various parts of the project. All information about the project are stored in a file entitled `Project.hyprj` (Hydratronics Project File) which is located in the project folder you created and named the project after. The name of the file `Project.hyprj` must not be changed as it is a file created by CANmaster PC-Tool. Each project has its own file with the same name.

## Open a saved project

CANmaster PC-Tool saves all information on the project in a file entitled `Project.hyprj` in the folder belonging to the project. By opening the project file in PC-Tool, you can use the built-in file manager to work on the project.

### Note

You can only work on one project at a time and each project must be located in its own master folder.

To open a project:

1. Select File -> Open Project...
2. Locate the file `Project.hyprj` in the project folder and click Open.

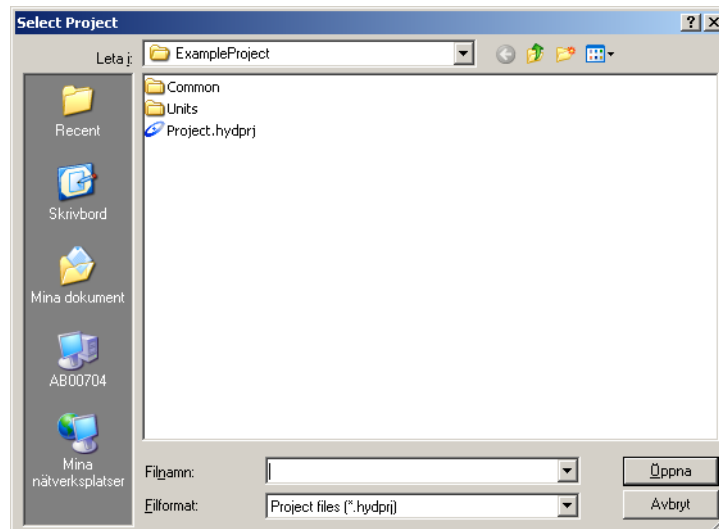


Figure 3.3. Common open project file dialog.

## Tip

Under the menu option File -> Reopen Project the five latest projects you have worked on are displayed. You open one of these by selecting a file name from the list.

## Files and resources

A project in CANmaster PC-Tool is organised in its own folder on your computer hard disk where all program files and other parts belonging to the project are collected in one place. The general information of the project is stored in the file `Project.hydprj` which contains a list of all the source code files, control units and all other information required to manage and build the project.

## Control units

A basic condition for creating a project is of course determining which control units to include in order to distribute the inputs and outputs required for the system to an appropriate number and type of control unit. The project you create as per the description above contains information on the control units included and how inputs and outputs are configured for each unit. The files that contain this information are created automatically by CANmaster PC-Tool. The files are located in the `Units` folder under the project's main folder. For each type of control unit (Unit) included in the project, there is a file with the control unit's name and the file extension `mtm`.

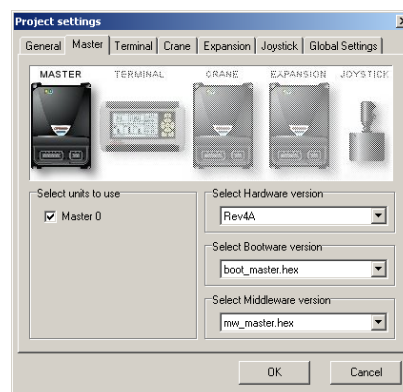


Figure 3.4. Project properties allowing to select and setup control units used in the project.

To enter or change the control units included in the project, do the following:

1. Select Project -> Properties.
2. Select a type of unit among the tabs. There is a tab for each type of unit. In addition to the types of unit there are also the tabs General and Global to handle other project settings.
3. Check the control units to be included in the project or remove the check to exclude units.
4. Set the version number of the hardware and version number of the base software for the units to be included.
5. Click OK to save the project properties and close the dialogue box.

### Number and type of control unit

The Master unit must always be included in a normal project by it having master control status in a CANmaster system and a system may only include one Master unit. There is one exception to this rule, when developing an application for an extended slave unit which will be described below. You can add or remove the other units in line with the requirement for the number and type of inputs and outputs for your machine application. A system may also only include one Terminal T2. For the other units such as Crane, Expansion and Joystick H5-S50, a maximum of *four units of each type* is permitted.

An extended slave unit is a crane or expansion unit which contains its own application software and due to this is able to run as a standalone unit without a master present. If such a project is to be created the master unit can be unchecked as soon as the correct slave unit is selected. An extended slave project can only contain one single unit and some functions usable only by the master unit is disabled. Other than that the application is created in the same ways as for a normal project.

A description of how you define and activate the inputs and outputs to be used for each control unit is included in the section called "Defining inputs and outputs".

## Source code files

The control program for the system is described in the different types of source code files. These include graphic source code files as well as text based source code files and table based configurations. The different file types uses different file extensions to make them easier to keep apart. Textual source files have the extension src, graphical diagrams is saved with extension hydsrc and CAN configuration files uses the extension hydcd.

To create a new graphic source code file:

1. Select File -> New -> New graphical source file.
2. Select File -> Save to save the source file.
3. Enter a file name and click OK to create the new file.

To create a new text based source code file:

1. Select File -> New -> New textual source file.
2. Select File -> Save to save the source file.
3. Enter a file name and click OK to create the new file.

To create a J1939 CAN node configuration table:

1. Select File -> New -> CAN node.
2. Select File -> Save to save the new configuration.

3. Enter a file name and click OK to create the new file.

To add a source code file to the project:

1. Select Project -> Add File to Project
2. Select the file type to be added from the sub menu.
3. Locate the required file in the file selector and click Add.

To remove a file from the project:

1. Select Project -> Remove From Project...
2. Select the files you want to remove.
3. Click OK to remove the files and close the dialogue box.

### Project browser file manager context menu

You can right-click in the project browser's file manager to bring up a context menu where you can also Add to, Remove from or Delete from project.

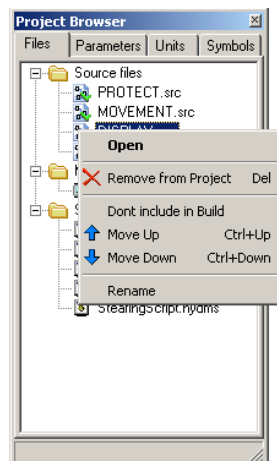


Figure 3.5. Project browser context menu in the Files tab.

### Note

When you remove a file using Remove from project the file does not disappear from the hard disk. If the file is to be removed completely you have to delete the file using Windows Explorer after you have removed the file from the project.

### Menu files for the terminal

The terminal's menu file is saved in a separate type of file with the file extension hydmd (Hydratronics Menu Definition). All settings and menus and the font data used are saved in the file. These files are listed in the project browser under the Files tab in the Menu files category.

To create a new menu file and add it to the project, do the following:

1. Select File -> New -> New menu source file.
2. Select File -> Save to save the new menu file.
3. Enter a file name and click Save to save the file and close the dialogue box.

You can create several different menu files for the terminal in the same project, for example for different machine versions, but you can only download one of these menu files to the terminal on the machine.

## Resulting program files

All the above mentioned files operate as input data to a compiler that translates the content of the files to executable program code in the Master Unit and Terminal. The executable files created must be downloaded in stages to the various units. The files contain basic information required to run the system. The main code for the application program downloaded to the Master Unit is created as a file called `out.hex`. In addition to this file, the file `out.hdr` needs to be downloaded. This file contains header information that is shared by the Master Unit and the Terminal in order for the terminal to know which variables are available and where they are located in the Master Unit. All parameters you have set are stored in binary format in the file `out.par`. A list is generated for the PC-Tool program of all symbols used in the system that are saved in a normal text file entitled `out.sym`.

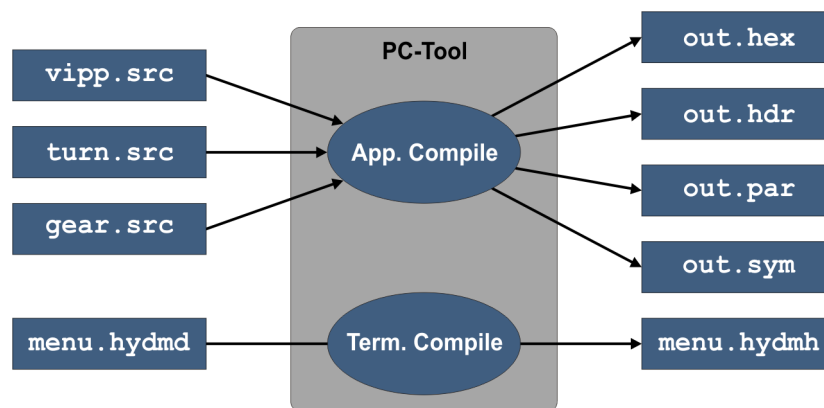


Figure 3.6. The different input files and the resulting binary files for downloading to the control units or to be used for testing the control program in the CANmaster PC-Tool program simulator.

Table 3.1. File types used in CANmaster PC-Tool projects.

File Name	File Type
*.hydc	CAN node configuration.
*.hydsr	Graphical source code.
*.src	Source code files for graphic and text based program code.
*.cmd	Script files for automatic control of simulations etc.
Project.hydrj	Main project file.
Units\*.mtm	Property files for the control units included in the project.
*.hex	Binary file ready to be downloaded to the Master Unit. This file contains the application program.
*.hdr	Binary file with information on the application program for communication with the terminal.
*.sym	List of all symbols included, their type and memory position.
*.hydm	Menu definition file that contains everything needed to compile the application specific terminal program.
*.hydnh	Compiled terminal program.
*.par	Binary file containing all the parameters of the entire project to be downloading to the control units.
*.can	Script or definition files for generating CAN bus packages for simulation and debugging of the second CAN bus channel in the Master Unit.



## Version management

### Product versions

A project always has a specific name that all management of the project is referred to as well as a version number. The version number is broken down into three parts, `major.minor.build`, which stand for Main version, Sub version and Build number.

To facilitate management of the versions there is the option of automatically incrementing an optional part of the version number each time a project is re-compiled. When a project is ready for testing on the machine, you set the version number under Project -> Properties... Then save the project and export it to another directory by selecting Project -> Export... Select a new location for the project in the dialogue box that appears. Select a directory with the same name as the current project but add the version number afterwards. This would result in a name like `MyProject 1.0`.

When you continue working on the project, you will be working in the normal working directory. If the control programs need to be updated on the existing machines, you can simply do this by opening the project with the version number that applies to the relevant machine and check the control program. Modifications implemented in a particular version do not affect other projects.

This type of version management gives you full control over the contents of the different versions of your control program as well as providing good historical records and order to the control programs. A good discipline is to create copies of the project for each new control program that is tested on the current machine type/series, and for each major version modification, and also create an external auditing system, a "Master" where all versions that are installed on the machines are saved separately from the PC-Tool directories thus avoiding unintentional changes to the installed version.

The version numbers entered for the project are also saved as a part of the control software that is downloaded to the control units. To minimise the risk of version conflicts and increase security for service work an automatic check is made each time the system starts to ensure all control units in the system have matching versions of both software and hardware.

### Version number

To set a project version number manually:

1. Select Project -> Properties...
2. Select the General tab.
3. Set the desired version number under Project Version.
4. Click OK to save the values and close the dialogue box.

CANmaster PC-Tool can also increment the version number automatically when you re-compile the whole project under the provision that compilation has no errors. This allows you to see immediately the versions you have in each downloaded unit and you know which of them is the latest. The three parts of the project are incremented in different ways according to the check boxes you selected.

**Build** The build count increments each time a project is compiled providing there are no compilation errors. A changed number for Build may mean that the control program has changed to some degree, but that no new parameters or symbols have been used.

**Minor** The middle part of the version number, Minor, increments automatically if the list of symbols or parameters has changed. When the value of a parameter changes, this is also counted as a sufficiently large enough difference in the program to increment this part of the version number. When minor is incremented, the build number is automatically set to zero.

**Major** The initial field of the version number, Major, is never incremented automatically but has to be done manually when it is appropriate for the project. By deselecting the check boxes for each

part of the version number, the incrementing of the version numbers will not take place when the project is re-compiled.

To automatically increment the version number when you compile a project:

1. Select Project -> Properties...
2. Select the General tab.
3. Select the parts of the version number you want to increment under Build Project. With a mark in the box this part of the version number is incremented.
4. Click OK to save the values and close the dialogue box.

To export a project to a new folder:

1. Select Project -> Export...
2. Select an empty folder or create a new one by clicking Create new folder. Do not select a folder where a project already exists, as this will be written over.
3. Click OK to copy the project files to the new folder. Note that only the files that you have added to the project are copied. Compiled binary files and other files, such as saved measurement data, are not copied.

## Version dependency

The modular structure of the control system gives benefits in handling and flexibility but also means that it becomes difficult to handle dependency between the different parts of the system. To handle this there is a number of control functions built into the various parts of the system. Generally speaking they can be divided into two parts; firstly the checks that take place in the control system at each start up of the system and secondly the checks that are carried out within the program CANmaster PC-Tool.

To be sure that all control units are able to communicate with each other and that all the utilised functionality is available in the system, a number of checks are carried out at each start up of the control system. To begin with, a check is made to see if the correct versions are being used in all connected control units. This is also done to ensure the correct functionality after replacing control units. The machine specific application program is then checked for compatibility with the underlying function library. This is done to compare the different tables of both used and available functions.

As there is also a strong dependency between the terminal program and the control program in the Master Unit, comparisons are made between name and version fields in these units. To benefit from the version number division, different parts of the version number are used for different comparisons. In order to start the terminal, the control program name has to be exactly the same as the terminal program. The name of both these programs is taken from the project name. Furthermore, both the 'major' and 'minor' parts of the version number must be exactly the same. The third and final part has the least significance, and in this case conformity is not required.

### Note

It is important to use the version numbers in the correct way in line with the description above to ensure the built-in control functions work. A modified value to the Build version number but with the same values for the other parts means that symbols and parameters have not been changed. However, if 'minor' or 'major' are not the same, there is no guarantee that the various programs can communicate correctly with each other.

In addition to the above mentioned checks that take place every time the control system starts, checks in CANmaster PC-Tool are also made before a program is downloaded to the control system. To avoid problems during operation, the same checks are carried out on the files that are to be downloaded to the units described above. For more information on how these checks are implemented and how any problems are reported, see the section called "Downloading application program and parameters".

## Revision control of source code files

CANmaster PC-Tool has no built-in support for handling source code versions and history over and above the manual procedure outlined above, but must be solved in a suitable way by the user of PC-Tool in the user's own organisation. The files that are interesting to keep tabs on are naturally the files you create yourself. The files that may be beyond the scope of the version management are all the files that are generated in some phase of the project. The following table lists the file extensions and whether they are suitable for version management or not.

Table 3.2. File types and usage of external revision control systems.

File Extension	Revision Control	Comment
src / hydsrc / hydcd	Yes	Source code file for control logics.
hydmd	Yes	Definition file for the terminal program.
hex	No	Generated binary file with compiled application program.
par	No	Generated binary file with parameter information.
sym	No	Generated text file with symbol information.
bmp	Yes	Image files that may be included in the terminal program.
cmd	Yes	Script file for test cases or check of simulations.
hydms	Yes	Script file for terminal program.
can	Yes	Script file for secondary CAN bus monitoring.
hydnh	No	Generated binary file with terminal program.
Files under Units	Yes	Definition files for the control unit properties and options. These are handled by CANmaster PC-Tool.

## Project delivery

This section describes how you proceed in order to deliver finished or updated projects to service personnel to enable them to update the systems out on the machines using their service computers.

To generate a data file that includes the update that can be stored in a portable data memory or be sent by e-mail, do the following:

Select Project -> Create Service Package. This will build the project and copy all necessary files to a zipped archive. If the operation succeeds the archive file can be found in the projects directory, it will be named after the projects name and version like: `MyProject_1_2_0_servicepack.zip`. This archive can then be unpacked by service personnel and used to update a CANmaster system.

A similar packet, only containing the source code files can be created by selecting Project -> Create Design Package. The resulting file will be called similar to the above example but having the `designpack` string in the name instead of `servicepack`. Thus, resulting in, for example: `MyProject_1_2_0_designpack.zip`.

# Design of application programs

---

## General

This chapter describes how you create an application program in CANmaster PC-Tool after you have made the basic settings for the project.

You normally begin by defining a number of inputs and outputs that you know will be used in the program. You do not need to define all I/O at this stage, as you can add and remove I/O over the course of the work. However, a framework of I/O must be created along with the control units they are to refer to in order to initiate an outline of the program structure.

This description starts by showing how you configure the inputs and outputs, and then how you use CANmaster PC-Tool to create the machine-specific application program through graphic design or working with text-based programming.

## Defining inputs and outputs

You select View -> I/O Designer from the main menu or click the I/O button in the toolbar to display the I/O designer window. A dialogue box pops up to help you configure the inputs and outputs.

### Symbol names

Your I/O should have descriptive symbol names that reflect the signal type and function. Instead of a name such as `ANALOG_4_IN` for an input that is connected to a pressure sensor to a hydraulic pump, you could name this input as `PUMP1_MAIN_PRESSURE` or something that gives a clear picture of the purpose of the input.

The symbol names can have up to 32 alphanumeric characters. The allowed characters are; A-Z, 0-9 and underscore "\_". A symbol name must not begin with a digit.

### Configuring I/O

The I/O designer has the following appearance:

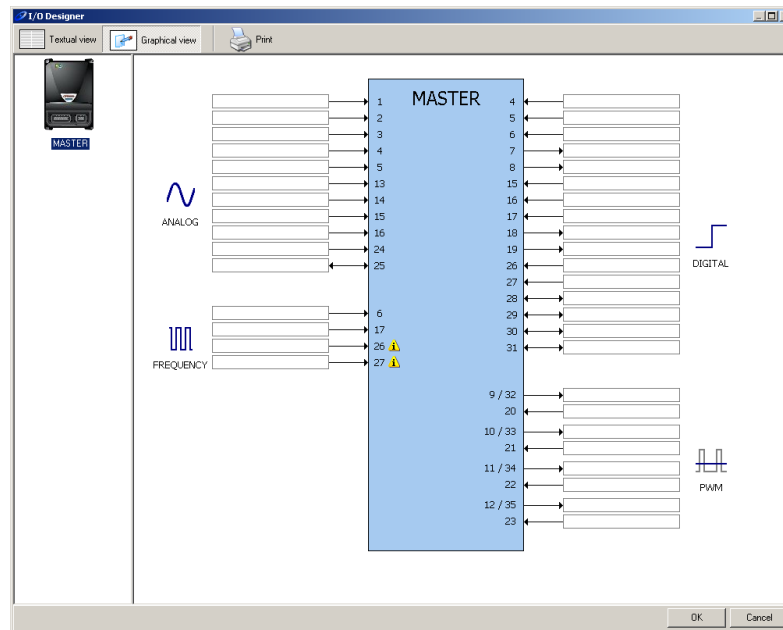


Figure 4.1. I/O Designer

To add a signal:

1. Double-click on the signal box where you want to add the signal.
2. An editor opens up in the box, enter the desired name of the new signal.
3. Press **ENTER**.

To change the name of a signal:

1. Double-click on the signal you want to rename.
2. An editor opens up, change the name.
3. Press **ENTER**.

To move a signal to another position:

1. Just drag the signal where you want it.

To swap two signals:

1. Drag one of the signals onto the other. When you drop the signal, the two signals are swapped.

To delete a signal:

1. Click on the signal you want to delete.
2. Press **DELETE**.

To change the direction of a multi-function signal:

1. Double-click on the arrow or right-click and use the context menu.

## Graphical programming

The graphical programming editor in CANmaster PC-Tool is a very powerful tool both for the design of control programs and for evaluating the control program.

By positioning ready-to-use function blocks from a library in a drawing area and linking the blocks with lines, you can quickly and reliably produce a control program without having any advanced knowledge in programming. The emphasis in this case is on the type of function you are looking to achieve, rather than on how you achieve it from a purely programming perspective. The graphic programming editor understandably gives a completely different overview of the program functions than the text-based editor. However, if you are used to working with text-based programming, it can in some cases be simpler to write shorter programming segments using the text editor that might otherwise require unnecessarily complicated graphical programming.

To open the graphic editor and start work on the control program, select File -> New -> New graphical source file.

## Tip

All source code files can be viewed in the project browser.

To change the name of a source code file:

1. Select the file you want to change the name of in the project browser.
2. Right-click the file name.
3. Select Rename from the context menu.
4. Enter the new name and press **Enter**.

## Programming window

When you open a new graphical programming window, two windows are displayed. The first window is a toolbox, the Graphical browser where you retrieve your function blocks and the other is the actual programming window where you place the blocks and draw the connections between the inputs and outputs.

You start your program design by selecting the required function blocks from the graphical browser and then dragging these with the cursor to the design area in the programming window. The function blocks inputs and outputs are then connected with lines in a similar way as when you draw the connection diagram between the components for an electrical or hydraulic system. The connection lines are created by dragging the cursor from an output to an input or vice versa. When you press the mouse button with the cursor placed on an output, you create a link in the form of an "rubber band" that is fixed at one end to the output and at the other end to the cursor. The end of the rubber band follows the cursor until you release the button on a valid input. A connection in the form of a line is then created between the blocks that is automatically drawn as the shortest route without crossing any function block. If you move a function block in the programming window, all the connections created will automatically follow.

## Function blocks

The whole graphical programming is about dropping the functions you need on the graphical code editor, and then connecting and configuring these to get the desired functionality.

A function block is drawn in the programming window as a rectangle with a black border. Each rectangle corresponds to a function or computation in the control program.

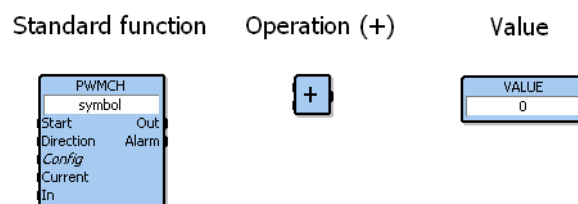


Figure 4.2. Example of different types of function block.

The reference describes each function block in detail.

The type of function is always shown in plain text at the top of the rectangle (see PWMCH and VALUE) or by a symbol in the middle of the rectangle (+).

Some function blocks have an edit field. Depending of the type of function block; a name, a value or an expression must be entered.

The inputs to the function block is drawn on the left. When the text on an input appear in italics, the input do not need to be connected as this have been assigned with a default value. For example, the PWMCH function block has one such input; *Config*.

The outputs from the function block is drawn on the right.

### Different types of function block

In principle there are four types of function block:

1. Fixed function blocks are simple computations that are converted to instructions directly in the control program. These could be simple arithmetic computations or logical comparisons. The function of the block cannot be affected in any way or be named and they can be used an optional amount of times in the same program.
2. Named function blocks include more advanced functions and are a part of the CANmaster system function library. Here each function block must be given a unique name to enable its use. The name is used to differentiate the parameters that belong to different function blocks of the same type.
3. Expression function blocks operate in the same way as a Fixed function block, with the addition that an expression must be specified for the block used in the program execution. Examples of such expressions are conditional expressions that choose the parts of the control program to be executed. Other function blocks that require different types of expression are selection functions and repeating functions.
4. I/O block: I/O blocks are used to access data in memory or constant values.

### Graphical objects

The `Graphical objects` is a window that contains all the function blocks that you can add to your design area in the programming window. As the number of function blocks is large, they are grouped according to type of functionality.

#### Tip

If you are unsure of the function of any block, you can select the name and press function key **F1** to get more information.

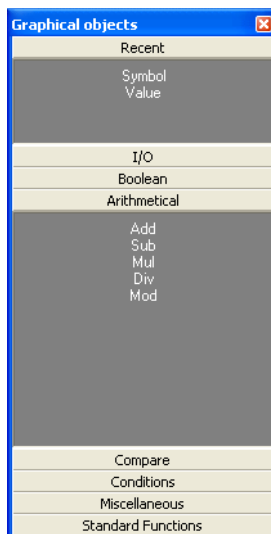


Figure 4.3. The graphical objects

The functions are named under each category and you use them by dragging the selected function with the cursor to the required position in the design area.

## Grouping

You can group several function blocks together.

To group, select the function blocks you wish to group and right-click on one of the blocks. Select Group Items in the menu option list. All the grouped function blocks are then drawn with a green border instead of a black one. When the function blocks are grouped, all function blocks move together when relocated.

To ungroup, right-click again on one of the function blocks in the group and then select Ungroup Items in the menu option list. All function blocks in the current group are then drawn in a normal way with a black border and the group is broken up. If you have created many minor groups of function blocks, you can remove the grouping for all blocks at the same time by right-clicking on the design area and selecting Ungroup All.

The grouping of the function blocks only applies as long as the programming window is open and is not saved in the program file. The function is only planned for use as a simple aid in your layout work and is only used to relocate the function blocks in the design area in order to give a better overview. A grouping of function blocks as per the description above has therefore no impact on the functionality of the control program and should not be confused with the handling of function blocks for grouping of functionality within a design area.

## Viewing connections

As previously mentioned, inputs and outputs are connected to the function blocks by dragging an "rubber band" from the input to the output or vice versa. Connections are automatically drawn in straight lines where they fit best without crossing other function blocks. They also follow when you move your function blocks around ensuring all lines remain intact. Even though all lines are always visible, there can easily be many parallel lines that also cross each other and it may be difficult to follow individual lines. The highlight function is a useful aid for checking the connections between the various function blocks. To display a connection, right-click with the cursor on an output or an input and, in the quick menu that displays, select Display Connection.

## Order of execution

Your application program is stored in the Master Unit and the program instructions are always implemented in a particular order depending on the structure of the program. The program's processing of



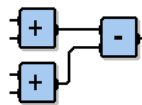
instructions is called execution. This execution always takes place sequentially and you have to determine the order of the various function blocks to be executed to achieve the desired program functionality.

The order of execution must be specified at different levels, partly the local order in each program file and partly the global order, i.e. the order in which the various global program files that are included in the control program are to be executed. Below is a description of how the order of execution is determined locally in each program file and then the order of execution from a global perspective.

### Local order of execution

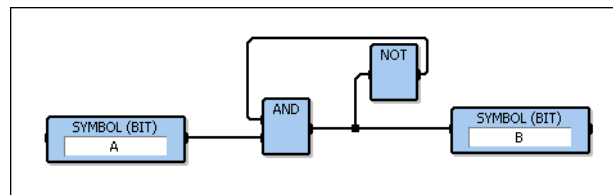
PC-Tool will execute the function blocks sequentially in the logical order. The logical order is defined as: if the result of block A is needed by block B, then block A will be executed before B.

In the figure the additions will always be executed before the subtraction, because the subtraction needs the results from the additions.



The simplest possible function blocks are used in the example to demonstrate the order of execution. The complexity of the function blocks makes no difference to how the logical order is calculated.

The next figure shows a so-called feedback. The AND will be calculated first and then the NOT will be executed. The AND will use the old value from NOT (the connection from the NOT to the AND is a feedback).



So what does this network do? When A is one, a pulse train will be generated on B.

To be able to check the final order of execution, you can right-click and select Display execution order in the popup menu.

### Global order of execution

The order of execution between different program files is totally governed by the order the files have in the project browser under the Source files folder. The top program file executes first. See the section called "Source code files". You can adjust the global order of execution afterwards by moving the program files in the project browser with Move up and Move down. See also the section called "Project browser file manager context menu".

## Configuring parameters

Different function blocks have different parameters that can be configured individually for each function block. By right-clicking on a function block and selecting Set Parameters... from the menu options displayed, you can configure and adjust the parameters that are defined for this particular function. For a more detailed description on parameter management, see the section called "Parameter editing".

## Data types

In the underlying control program, different data types are used to handle the values of the signals. Depending on the type of value used to compute and the type of function they are used for, the correct data type must be used. For graphical programming the data type is calculated automatically as far as is possible, but sometimes there may be a reason to specify the data type manually.

By right-clicking an output on a function block, where it is possible to specify the data type, you can select Configure I/O from the context menu. The dialogue box displayed allows you to select the size of the data type and to specify whether it can have both positive and negative values, or only positive.

You can remove the data type set manually for a configured output using the same dialogue box by clicking Clear.

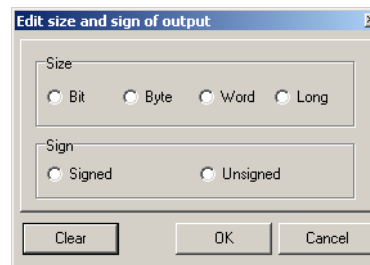


Figure 4.4. Specify the data type for an output on a function block.

To view the data type that has been assigned to a particular connection line in the design area, either manually or automatically, you can right-click in the work space and select Display size and sign. Note that a data type that has larger values requires more memory and processor time to compute and to handle which is why it is always wise to use data types that are as small as possible.

The handling of data types mentioned above only applies in each program file. When all files in the project are brought together to the complete control program, the largest of the data types is selected for each symbol. In this way, no information will disappear because of a symbol being defined as a byte in a file but as a word in another, in which case the symbol will be treated as a word in the whole control program.

## Text-based programming

Everything that can be produced with the graphical programming editor can also be created using standard text-based programming. There is also support for this working method in CANmaster PC-Tool as it can be a lot clearer in certain cases.

The source code is written in an integrated text editor that has a built-in syntax display with different fonts and colours depending on the significance of the text in its context. This makes the source code easier to read and write.

To create a new source code file and add it to the project, select File -> New -> New textual source file or click on the New button in the toolbar and select Source Code.

As using the text editor requires very good prior knowledge in basic programming techniques, this is not described in detail in this manual. It might be worth mentioning that there are a number of different settings that can be made to get the text to appear in the way that suits you best. Right-clicking in the text area in the editor and selecting Properties allows you to adjust the length of the tabs, colours and other normal settings.

## Compiling program files

To convert the program files that are designed and grouped in the project browser to an executable program that can be downloaded to the control units, you must compile all source code to binary format. Compiling uses source code files as input data, both graphic design and text based source code, that you have created and convert these to the format that can be run directly in the control units. During compilation, checks for syntax errors in the program are made and that all symbols and other variables are placed in the memory in the control units.

You can compile each source code file individually. The results of the compilation are shown and you have the option of fixing any errors. This is done by right-clicking and selecting Validate diagram in the popupmenu.

In the graphic program compilation, errors are marked directly in the design area. As an example, an input that is not connected is marked red. You can then clearly see where the errors are and can correct them straight away. When all syntax has been generated correctly as the last step before compilation, it is often only misspelled symbol names and inputs that are not connected that can be wrong. The exact error message is displayed as a tool tip when you move the cursor over the marked error. Correct the error and try to Compile again.

In the text-based case, all compilation errors are displayed in the log window, where an error code is displayed together with the file name and row number. Double-clicking on an error message marks the offending row in the text editor and you can immediately correct the error. If you need help interpreting what an error message means more in detail, you can mark the error message and press function key **F1**, which will bring up a more extensive help text that describes the error and provides examples of what the fault might be and how you can correct it.

## CAN Configuration

The common usage of J1939 CAN communication in machine control systems are made available in the CANmaster via a table based CAN configuration. The CAN configuration allows for configuring any number of J1939 CAN nodes and which messages to exchange between the CANmaster system and the J1939 node. These nodes can be such things as diesel engine controller or gearbox controller or similar.

In a CANmaster that uses J1939 communication each peer on the CAN bus will be represented by a separate configuration file. Each configuration file holds the messages used for that specific node. In this chapter the CAN configuration will be introduced and it will be shown how to use.

### Nodes

The CANmaster control system looks at the external J1939 as separate nodes. One such node can be the diesel engine controller or a gerabox controller for example. What distinguishes nodes are their source address. Each node has a unique source address that the CANmaster system uses to select what the messages it receives actually are.

For each node the CANmaster system can have different source addresses, that is, the address the CANmaster system uses when it sends messages to the nodes. Moreover, each node is named to keep them separate and allows for better symbol name generation. The following shows the user interface for setting this information. One such view will be available for each node in the project.

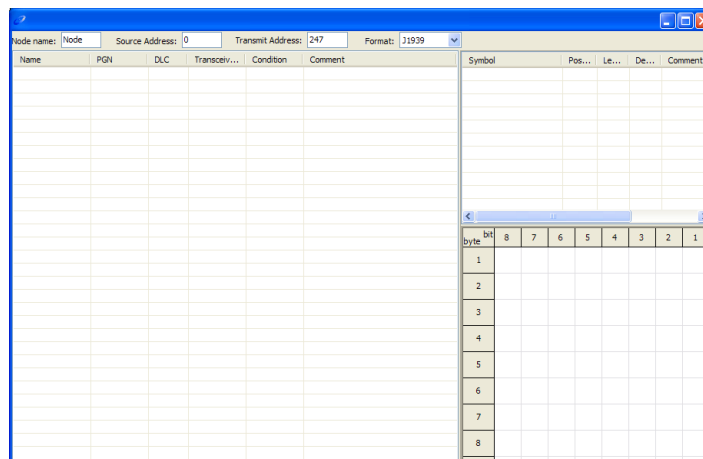


Figure 5.1. The main CAN configuration window.

At the top of the window the node name, source address of the messages sent from this node, the source address of the CANmaster used when transmitting messages to the node and a CAN format specifier. The format is reserved for future use, currently only J1939 is supported. Below the node specification in the header row the window is split into three main parts:

Message list	The largest part of the window is the message list. This holds a list and specification of each message used in communication with each other.
Message data list	When a message in the main list is selected all its data parts are listed in the upper right part.
Data layout	A visual help to show the data parts in the selected message.

## Messages

The core of the configurator is the message list. A message is the packet of data that is sent on the CAN bus between the CANmaster system and the external J1939 node. A message is named to be able to refer to it internally and during monitoring. On the CAN bus, the message is identified using the PGN, Parameter Group Number, which is defined in the second column. The DLC then is used to set the length of the message. Each message can be of certain transmission type, basically it is either sent or received by the CANmaster system. To enable some application specific logic a message specific condition can also be specified. Finally a comment allows for some documentation.

The following sections will reiterate the different columns of the message list in more detail.

### Name

The name of the message is used to keep track of the messages and be able to refer to the messages in a well defined manner. The names must be unique in the system. The name will be used to generate several internal symbols that are used to handle the logic regarding reception and transmission timeouts and such things. It will also be used in messages regarding misconfigurations or similar.

The checkbox before the name is used to activate or inactivate the message.

### PGN

The PGN is the the identification used for the CAN message on the bus. This is built from the J1939 specified PF and PS fields together with the page and reserved bit. In the main list they are shown as the PGN number. By double clicking on the cell in the table the dialog shown below is opened to help define the PGN number.

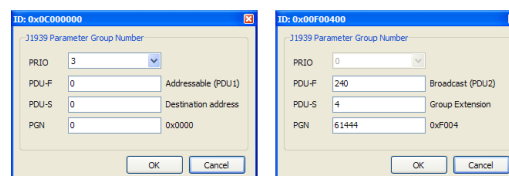


Figure 5.2. Dialogue to define the PGN number for messages. For tx messages the priority can be set but not for rx messages.

The dialogue for setting PGN allows for setting it by defining the PF and PS separately or by setting the PGN directly, in which case the PF and PS fields are updated automatically. A little hint is the label on the right of the fields that states the type of PF and PS that has been entered.

### DLC

The length of the message must be specified to be able to handle it correctly. In J1939 most messages are of size 8, that is a standard CAN message. However, using the J1939 Transport protocol messages can be larger than one standard messages, and also of variable size. To handle larger messages, CANmaster allows specifying a DLC that is larger than 8. This will automatically use the transport protocol to send and receive these messages.

For variable size messages the maximum size of the data handled in the CANmaster system must be given since there is no way to dynamically handle memory and symbols. When specifying the size of a

message they are defined of variable size if the checkbox is checked and will be presented as VAR (x) when the size is set to x. In this case the x is the maximum size handled by CANmaster, all data above this limit will be discarded.

## Transceive Method

Basically, a message can either be sent or received. This is as viewed from the CANmaster system. A message to send from CANmaster to a J1939 node is set as tx while a message to be received will have the transceive method set to rx. Additionally, some of the J1939 messages uses a special PGN message to request a value from another node. By setting the transceive method as rx request the request message is generated automatically and everything else is handled as a normally received messages.

The transceive method has influence on the way the PGN editor is shown as can be seen above. It also affects the available conditions in the condition column.

## Condition

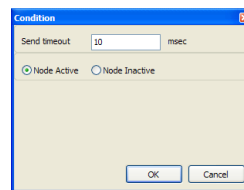


Figure 5.3. The condition editor.

Depending on the transceive method set for the message different conditions can be added. The different types of conditions are:

Node Alive	This condition is used for transmitted messages. If the Node Active is chosen, the message is only transmitted if the node has sent some messages and no expected message from the node has timedout. The condition can also be inverted to only send message when the node has timedout or not sent any messages that are received and handled by the CANmaster system.
Send Timeout	The send timeout condition is a timer used to set an interval for sending the message. The timeout is set in milliseconds, but the lowest resolution used is 10 ms.
Receive Timeout	The receive timeout is used to flag the node alive or not. When a message has a receive timeout set and the node is not sending the message within this interval the dataparts of the message will be set to default values and a flag for the node will be reset to indicate it is no longer available on the bus.
Condition	The condition is a general condition that allows the application to set some symbol value that is checked before the message is transmitted. This condition allows for different types of common comparison checks.

The condition is mainly used to set the transmit and receive timeouts. Also, the Node Alive condition is commonly used for not sending out messages on the bus unless the peer node is available. The following table indicates which conditions are available for each transceive method.

Table 5.1. Available condition usage in relation to transceive method.

Condition Type	rx	tx	rx, request
Node Alive		✓	✓
Send Timeout		✓	✓
Receive Timeout	✓		✓

Condition Type	rx	tx	rx, request
Condition		✓	✓

The conditions are added via the context menu. Choose the wanted condition from the Add sub menu.

### Comment

The comment field provide a simple way for documenting the message.

### Message content

When a message is selected in the main message list all its defined contents will be shown in the top right part of the configuration window. This part is a list of CANmaster symbols and their mapping to the CAN message sent or received. A datapart of a message can be either active or inactive as indicated with the check box in the first column. When inactive, no symbol or memory will be used for handling that part of the message. This can be used to fill in the full message specification even though not all parts are implemented.

The Symbol column is used to define the name of the symbol used in the CANmaster application to refer to the data sent or received in the specific message. The value of the fields in this column must adhere to the general symbol name guidelines. The name must also be unique among the configured symbols for CAN configuration. In the application logic, the symbol can then be used as a regular symbol but there will be a type error if a manually set datatype differs from the datatype used for the CAN configuration handling.

The Position column defines the position of the data in the CAN message. The value has the format `<byte>.<bit>`.

The Length field defines the number of bits in the message the data part is using. The data members of the message must be in a consecutive part but can be place over any byte boundaries. The length can be any number between 1 and 32. The symbol created will be of the type with the closest match that can hold all data. For single bit values the BIT datatype is used. Values of length 2 - 8 will be of type BYTE and lengths 9 - 16 will result in a WORD datatype. Any size above that will result in LONG datatype.

The Default column is used to set a default value of the symbol in CANmaster. This value is used if a receive message has timed-out or before it has been received the first time. For transmitted messages the default is only used if the symbol is inactive. In that case, the default value is sent in the place of the inactive symbol.

The last column provides a simple documentation comment.

All undefined bits in a message will be filled in with binary ones. To make it easier to spot these parts of the message and to get a good feeling of what parts of the message is used and not the bottom right pane will show the data parts layout visually.

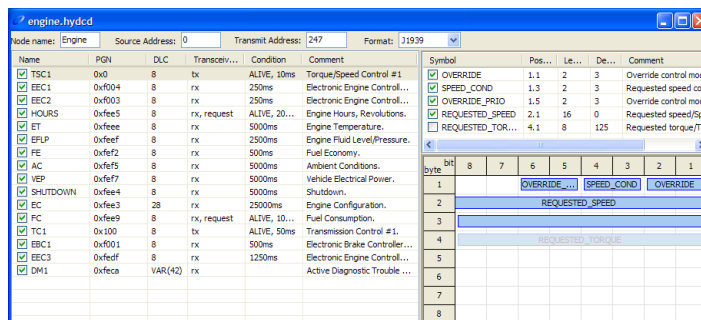


Figure 5.4. Some filled in data in the CAN configuration.

## Browsing standard messages

To make it easier to use the J1939 CAN configuration the most commonly used CAN messages are already provided to be simple added to the application. The message browser is shown by selecting Browse Messages... from the context menu in the message list.

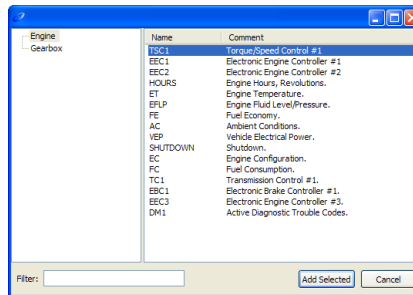


Figure 5.5. Standard message browser.

A dialog is shown with all commonly used messages predefined. The window is split in three panes. The left main pane allows for some grouping of messages. We have chosen to group messages most commonly used for different types of ECU:s such as engine controllers or gearbox controllers. By selecting one of these all messages will be listed in the right pane with a short name of the message and a comment.

By hovering the mouse over one of the messages in the list, a tooltip will show all the data parts of the message. This makes it easier to select the correct message. Another helpful feature of the message browser is the filter in the bottom pane. With the filter text box, it is possible to show only messages where message name, comment or data parts names and comments matches the text in the filter box. This can greatly reduce the manual searching through the list of messages if one knows what is needed.

For example, to find the message used for sending the current fuel consumption one would enter `fuel` in the filter text box. The list of messages will be reduced for each letter typed until only a few messages are left in the list matching the filter term. It is now rather easy to choose the right message by manually inspecting the message, select the wanted messages and click on Add Selected.





# Terminal Design

---

## General

The Terminal represents a vital part of the machine's control system. The terminal is used in the daily work for viewing the machine's operating data, for selecting operating mode, acquiring information on faults in the system, and for adjusting parameters and calibrating sensors. Simpler types of troubleshooting of the control system can also be done when a service computer is unavailable, such as logging of values for the inputs and outputs and retrieving data in the central error log of the system.

It is therefore important that terminal management is designed as simply and as user-friendly as possible for both the machine operator that use the terminal every day and for the service personnel so they can quickly carry out basic adjustments to the machine system.

Terminal Design is the integrated programming tool in the CANmaster PC-Tool concept that gives you a free hand in designing the various functions for the Terminal, where menu structures, use of buttons, built-in functions in the form of terminal scripts, texts, images and symbols for screen display can be customised to suit the type of machine, user group and application

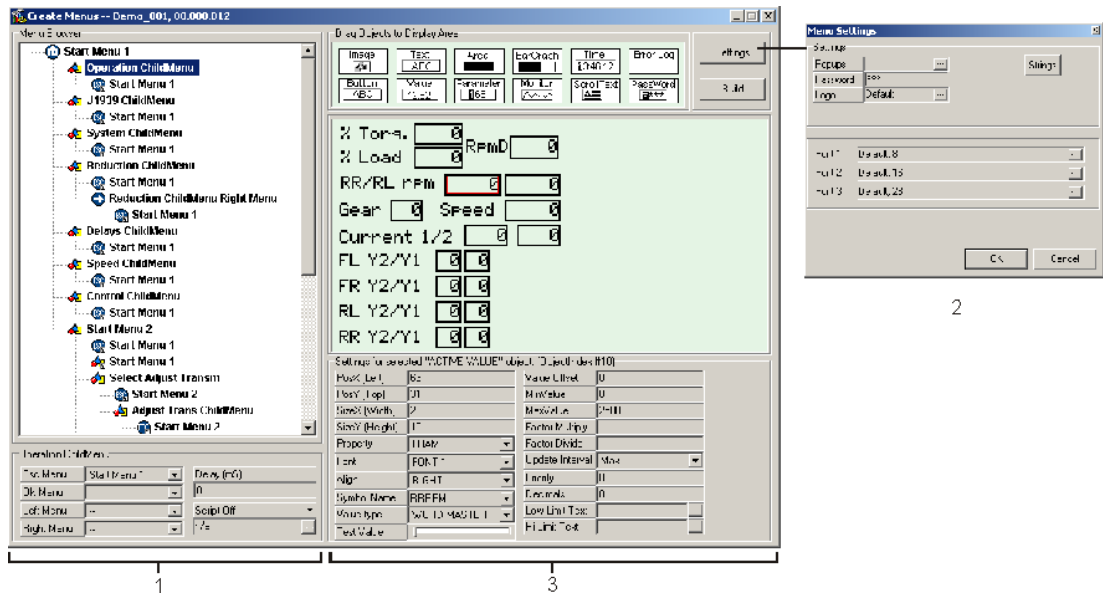
Terminal Design can be regarded as a separate programming editor in the same way as the graphical editor used when designing the application program or the text-based programming editor.

The programming tool is of the 'what you see is what you get' type, where you design menus on your computer screen while you see the results in real time of how it will work in reality with the machine's Terminal.

Creating a new empty program file for the terminal and adding it to the project is described in the section called "Menu files for the terminal".

## Main window in Terminal Design

Terminal Design contains a number of aids for designing menu structures and screen visuals.



The Terminal Design tool window is divided into three main sections as per the figure above:

1. The structural part that displays the structure and layout of the menu tree.
2. The global settings that apply to all menus in the menu tree.
3. The drawing area with various tools for design of the screen visuals.

### Menu structure and navigation

The menu tree always has at least one menu called the root. The root menu appears at the top of the tree view and represents the first menu the user sees on the terminal display when the system starts.

The name of each menu is displayed in the tree view. The default name of the root menu is specified as "Start Menu", but this can easily be changed by right-clicking the name and selecting Rename. You can also select a menu and then left-click again to give you the option of directly changing the menu name.

To change the name of a menu:

1. Select the menu you want to change the name of by clicking it in the list.
2. Select Rename from the context menu that appears when you right-click the name.
3. Enter the new name and press the **Enter** key.

A home icon appears in front of the root menu name in the form of a house symbol. This means it is the menu tree's start page. The icons further down in the tree show the buttons that lead to sub-menus directly under a main menu. In this way you can clearly see what the user has to press to navigate through the menu tree. In the same way, the menu tree continues to grow until a button leads back to a menu that has already been created in another part of the menu tree. The button icon and the menu name are displayed but it cannot have more sub-menus when these are put in their original places.

This allows the creation of shortcuts and the building of logical structures that the user can easily navigate in. The aim is that the Terminal menus are as easy to use as possible and that the user can quickly find the required information.

Under the menu tree window there are navigation properties for the selected menu. You select a menu by left-clicking it. The selected, active menu appears with a blue background in the menu tree. The user of the terminal can move between the menus by pressing the various navigation buttons on the terminal.

## Buttons for navigating between the menus

### Terminal buttons

There are four buttons on the terminal that have particular importance when navigating between menus:



- Home
- Back (Escape)
- Right arrow
- Left arrow

You can change the function of the buttons in the selected menu by deciding what happens when the user presses one of these buttons. The navigation properties have multi-option lists for each one of the four buttons. By selecting a menu in one of these lists for a particular button, you can set the menu to be displayed when the user presses a corresponding button on the terminal. It is also here you create new menus by selecting the Create new menu... option.

To create a new menu:

1. Select the menu you want to use as the base menu for the new menu design.
2. Select Create new menu... from the list to the right of the button that you want to lead to the new menu.
3. Design the new menu that is automatically made active.

### Soft keys

In addition to the four standard buttons that control menu navigation, you can also move between the menu screens using 'soft keys' that you position on the terminal display. The user moves between the soft keys using the ARROW UP and ARROW DOWN buttons on the terminal and activates the active button with the OK button. This lets you place several shortcuts on a menu screen and you can control a large part of the terminal management via 'buttons' on the terminal display.

If you add menus using soft keys, you cannot connect the menus to the OK button at the same time as it is secondary to the soft keys.

To create a menu shortcut using a soft key:

1. Select the menu you want to use as the base menu for the new menu.
2. Drop a soft key on the screen by dragging it from the object browser.
3. Change the name of the button using the Text object property.
4. Specify the menu to be activated in the Child menu object property at the bottom when the user activates the button.

Another way of creating menus is Hidden menus. These menus only appear when a correct password has been entered and checked through the active password object. The password used is set as a global

menu parameter. The password may contain four characters and each character may be a figure 0-9 or a letter between A-Z. The user uses the ARROW UP and ARROW DOWN keys on the terminal to change each password character and moves between the characters using the right and left arrow keys. You create a hidden menu in the same way as a menu shortcut but you use a password object instead of a soft key. This type of menu is suitable to use for functions that are only available for authorised personnel from a security or application perspective.

Each menu has the option of holding a terminal Script. The terminal scripts are small program segments that the terminal executes when a particular menu is active. You can use these programs to automate tasks that otherwise would require many key strokes. You will find more information about terminal scripts in this chapter under the section called "Terminal script".

To connect a terminal script to a menu, do following:

1. Select the menu you want to connect to the terminal script.
2. Click on the file option button with three dots under the Script heading.
3. Select the script file that you want to connect to the selected menu.
4. Click OK to save the settings and close the file option box.

## Global settings

The terminal program has a number of settings that apply to all menus in the current project. These include global pop-up texts for warnings and alarms, the fonts that are used in the menus and the passwords for access to the service menus.

### Show a logotype at system start-up

When the system starts, the first thing that appears is an image to identify the system. The image may be a logotype, relevant machine type and machine serial number or similar item and is added to the other global settings. The terminal screen may only show images in black and white and is limited to images with a maximum of 240 x 128 pixels. If the image is in colour when it is imported, it is automatically converted to a black and white presentation. CANmaster PC-Tool supports bitmap images of the type BMP format that you can create in a standard drawing program. For the best results you should design the image in black and white with the relevant resolution. It is worth spending some time on the image as it will be used and displayed on the machine every day for many years to come. Avoid complex images and you should try to use images that are as small as possible to avoid loading the terminal's image memory unnecessarily.

To show your own logotype on the terminal when system starts:

1. Select the browse button by Logo from the global settings.
2. Select Add under the list of images in the terminal.
3. Locate the image file that contains the selected logotype. The image should be in black and white when imported with a maximum size of 240 x 128 pixels corresponding to the image area of the terminal.

### Fonts

In the menus' graphical objects that contain text, you can select from three different fonts of varying sizes. These three fonts are specified among the global settings under Font 1, 2 and 3.

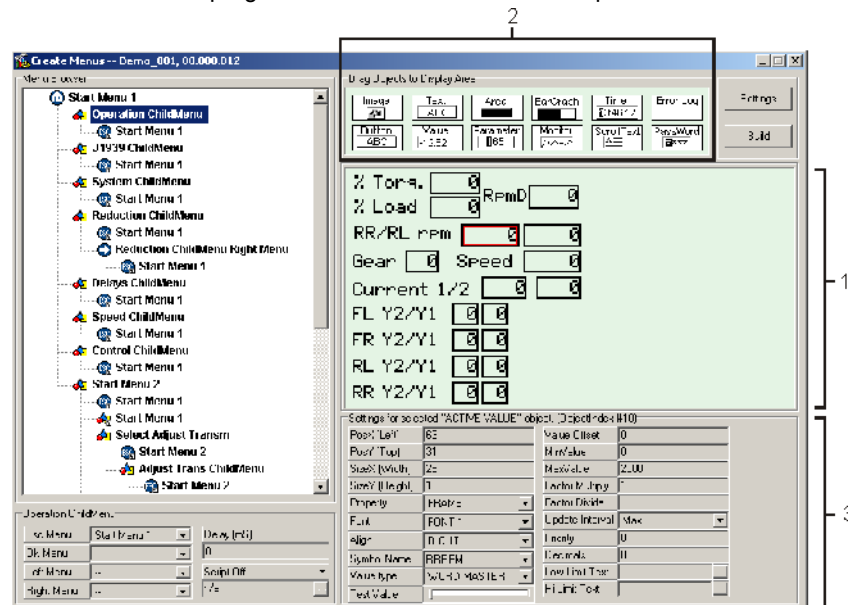
### Password

To set a password for service access, you simply enter the password in the text field to the right of the Password text. The password is hidden behind asterisks. The password is displayed on the terminal in

clear text as you enter it. The password consists of four characters and each character may be a figure 0-9 or a letter between A-Z.

## Screen visuals - general

The area for construction and design of what is to be viewed on the terminal display for each menu and any connections to the control program are divided into three main parts:



### 1. Drawing area

The space in the middle of the area shows the active menu's appearance in the Terminal display. You drag the objects from the object browser and drop into the drawing area to build up the screen visuals for each menu. The space corresponds to a resolution of 240 x 128 pixels.

### 2. Object browser

The object browser at the top contains the menu objects that are available for dropping into the drawing area.

### 3. Object properties

The object properties under the drawing space contain lists of properties and their values for the selected menu object in the drawing area. Each menu object has a number of properties that determine its design and function. These properties include place, size and the font that is used for text display as well as any connections to variables in the control program.

To view the properties of the objects that are placed in the drawing space, you select the object by clicking it whereupon a red border appears around the object. The relevant values for the selected object are shown in the object properties under the drawing space. By changing these values you can immediately see how the objects are affected. For the objects that are connected to measurement and control values, you can simulate the signal values by setting a test value using the slider at the bottom by Test value.

## Note

You can only change properties for one object at a time.

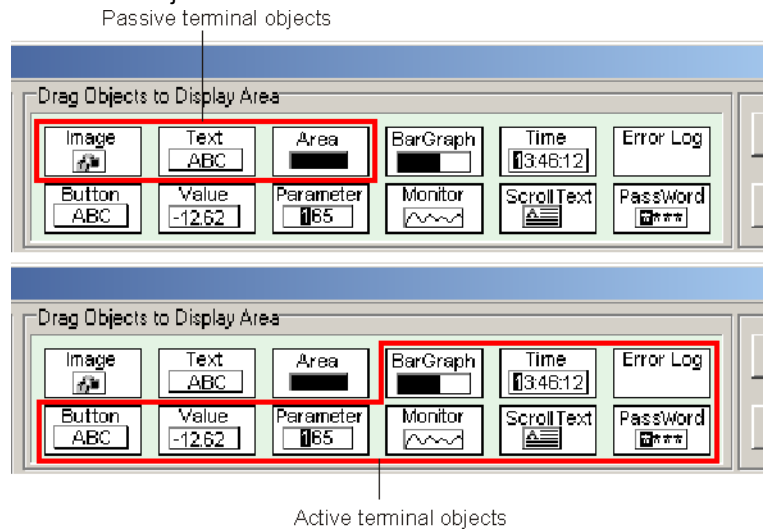
You can also change the location of the objects in the drawing area by selecting them and dragging with the cursor to the required place. The size of the object can be changed by dragging the object border.

The position and size of the object can also be specified by specifying the values for PosX, PosY, SizeX and SizeY.

The placement of the objects in the display area may overlap each other, in which case it is important to specify the order they are to be placed on each other. You can simply alter the order depth wise by right-clicking a selected object and selecting Bring to top or Bring to back.

## Terminal objects

The terminal objects are the elements in the terminal display that build up the menus and their functionality. The various terminal objects that are available can be divided up into two main categories; Passive and Active terminal objects.se



### Passive terminal objects

Passive terminal objects are used to give a structure or information to the machine operator. A passive terminal object is static and is not changed in any way when the terminal is used. Examples of these are images, labels and various lines or background areas. The option of hiding or displaying static terminal objects is available via terminal scripts.

### Active terminal objects

Unlike the passive objects the active terminal objects have the option of displaying and changing values in the control system via the terminal. There are terminal objects that can be connected to a particular signal value and display this as numbers or in the form of various graphical presentations. Other active terminal objects provide the machine operator with the option of going in and changing parameters or adjusting the system clock.

## Design of screen visuals

To design the screen visual for a menu:

1. Drop terminal objects in the drawing area by dragging them from the object browser over to the drawing area.
2. Adjust the properties of the objects in the property editor below the display window. Depending on the type of object there are different properties that can be set for the object.

### Note

The maximum number of terminal objects for a menu is 50.

## Passive terminal objects

Image	<p>Displays an image. An image must be of the BMP file type that you can create in a standard drawing program. The image you use must be black and white and should be as small as possible to save memory. Remember that the screen is limited to 240 x 128 pixels. The properties of an image consist of a position that specifies the image's upper left corner which corresponds to the coordinates (X, Y) = (0, 0). The X coordinate increases to the right of the screen while the Y coordinate increases as you move down on the screen.</p> <p>In addition to the position, the image has properties for how the border and background appear. These properties are shared for all types of terminal object.</p> <p>All images in the terminal are stored in a list. When you insert an image you automatically create a reference to it from the built-in list thus ensuring the same image can be used in several places without overloading the memory. New images to be added to the list are retrieved via the dialogue box for image selection using the Add command and are automatically added to the list. The new image is compared with the existing images in the list to prevent identical images being stored in the list. You specify the image you are referring to under the Picture property. By clicking the button with three dots in the row Picture property, the dialogue box for image selection appears giving you the option of adjusting the image list or selecting another image. The dialogue box also shows the amount of available image memory that has been utilised.</p>
Static text	<p>Shows a text that cannot be changed by the machine operator. The text field can contain a maximum of 30 characters and is limited to one row. The normal properties for text placement and size are available together with display properties such as border and inversion. The properties that are specific to the objects that contain text are the font to be used and the alignment of text in the specified text field. Here you can select from left alignment, centering or right alignment. Finally, specify the Text property at the bottom under Settings for what is to appear in the text field.</p>
Area	<p>A static area that can be used to create lines or backgrounds. In the same way as all other terminal objects there are properties for placing the area object on the menu screen. Placement and size are the main properties for this object. In addition to this there is the option of showing the area with a border, flashing, inverted or normal with black colour.</p>

## Active terminal objects

Button	<p>A button (soft key) that the machine operator can use to activate a function. Buttons have properties that match the text field. In other words you have the option of placing and changing the size, setting display alternatives, fonts and text adjustments and the text to be displayed on the button. In addition to these properties there are other properties that link the button to another menus.</p> <p>You can use the buttons to navigate more efficiently between menus than just using the standard buttons on the terminal for navigation to the left and right. The user can choose between several buttons that you have placed in a menu by moving between them with the arrow up and arrow down keys. The active button is highlighted and if the user presses OK the menu is activated that is associated with the active button.</p>
Value	<p>A text field that shows a signal value or other type of variable that is updated from the system. When a value field is a text field, the same properties as for a text field are also available for a value field apart from the text that is displayed. In addition to the normal properties, there are properties that link a value field with a symbol in the control program. Property Symbol name is that particular property. In order for the terminal to know what type of value is involved, you specify the Value type property.</p> <p>By adjusting the Test Value property you get an idea of how the object will appear in the real terminal when the value is changed.</p>



## Note

The test value is just intended for test purpose and is not saved to the terminal program.

You can set additional adjustments of the value for presentation of the terminal display with the properties Value Offset, MinValue, MaxValue, Factor Multiply and Factor Divide.

Using the Update Interval property, you can specify how often the value is to be read from the Master unit. By selecting the Max, the value is read at the maximum frequency. Avoid using the maximum update frequency as this could mean a dramatic increase in the load on the CAN bus if several values in a menu are to be updated simultaneously. Select the lowest update frequency that is required for each individual case.

The total number of menu objects of the Value type is 32 for a screen page.

**Parameter** A text field where the machine operator can modify the contents. The properties for the parameter object are similar to those of the value object. You can specify the location and size of the parameter object as well as the font and display options that are to be used.

The Auto Update property can be set as On or Off. When it is set as On, the value is automatically updated from the Master unit after two seconds, whereas in the other case the user must first press OK to update the parameter value. As updating can be demanding for the CAN bus, you are recommended to use Off in normal cases.

A maximum of 10 parameter structures can be edited for each menu screen. However, a structure may include several parameters and all of these can be edited together. Depending on the type of structure, the number of parameter objects may differ. The limiting factor is the Buffer Number property where each structure must be saved in a separate buffer.

The Param Lowest and Param Highest properties specify the range for the adjustment of the parameter, and the Change Step property specifies the size of the changes made when the user adjusts the value with the arrow up and arrow down keys.

The properties for the parameters to be adjusted are divided into two parts. Firstly you specify the function the parameter is connected to and secondly the function parameters you want to adjust. Select the function in the upper list first and then the parameters to be adjusted in the lower list.

**Bar Graph** A bar graph that shows the graphical appearance of a value for a variable. The Bar Graph has the majority of properties that correspond to a value object. There are placement, size and display options. The difference between the value object and bar graph is that the latter does not display any text. The direction in which the bars are to be drawn is shown instead of text. The bar direction is specified using the Align property. The other properties for adjusting the value prior to display on the screen are the same as for the value object.

**Monitor** A diagram that shows the graphical appearance of a variable's change in value over time. The value shown here has a history that constantly varies with the passage of time. The properties for placement and position are as they are for most other terminal objects. The display options are also the same. You select the value that should be presented in the Symbol Name property and the type in the Value type property. You can adjust the value before the presentation on the screen just as you did for the bar graph and value object with an offset, minimum and maximum value and scaling in line with simple arithmetic factors. The size and position for a monitor is limited to even eight pixels. This means that monitor objects can be placed at the coordinates 0, 8, 16, 24 in each direction in x- and y-axis.

Scroll text	A static text field that shows text that may be several rows. This terminal object can be used to show help or information texts for the machine operator or service personnel. If the text is longer than the screen display area the user can scroll using the arrow up and arrow down keys. The properties for this type of object are like the ones for the simpler text object. However, you have the option of entering much longer text parts divided over several rows using this object. You decide where the line breaks are to be in this object. For multi-language support, you translate line by line, which means that the number of rows for each language must be the same. If you discover that the number of rows for one language has to be more, you can add empty rows in the default language to provide for more rows in the other language.
Time	Shows the current time and allows the user to update the system clock. The properties for this object are very simple. Only placement, size, display options and fonts can be set.  The current time in the system is displayed for the user and by activating the object, the user can reset the clock one field at a time. The user moves between the time fields via the right and left arrow keys, and adjusts the value using the arrow up and arrow down keys. When the new time has been set, the user then presses OK to update the system clock.
Error log	The central error log in the Master unit can be retrieved and viewed on the terminal. The last registered error appears at the top of the list. If more space is needed for the error messages the user can scroll the list by using the arrow up and arrow down keys. The only properties that can be set in this terminal object are position, size, display options and fonts.
Password	A text field with protected data entry that can be used to restrict access to certain menus that require a certain level of authorisation. Properties for the password object are position, size, display options, fonts and the menu screen that is displayed when the user enters a correct password.  The user activates the password object by pressing OK. The user can adjust one character at a time using the arrow up and arrow down keys and move between the characters in the password with the left and right arrow keys. The three characters that are inactive are displayed as asterisks.

## Icons and fonts

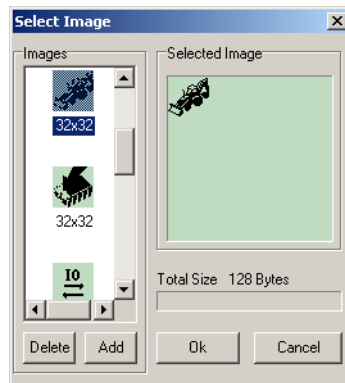
The terminal management and screen display should be designed as uniformly and user-friendly as possible to allow the machine operator to quickly find the machine's various functions and to be able to make the adjustments necessary in a safe way. The tools that can be used for this include the graphical expression of selecting fonts and distinct icons.

The terminal's graphical object has the option of selecting the fonts to be used. You have a free hand in selecting three of the fonts installed on your computer. You select these by clicking the font selector next to the name of the font, i.e. Font 1, 2 and 3 from the global settings. In the terminal objects that use text, the Font property is available to help you choose one of these. The fonts can be used for different text sizes for headers and measurement values where extra large fonts can be used to show the most important measurement values. Remember that the larger the font you select the greater the memory used in the terminal.

In addition to the option of using different fonts to highlight different types of information, there is also the option of using images or icons. An icon in this context is the same as a small image. All images used must be black and white screen images in BMP bitmap format.

When you place a terminal object of the image type, a dialogue box appears where you have the option of selecting the image you would like to view in the terminal display. A list of the images that have been entered is shown in the dialogue box. Press Add to add new images to the list.

When you have selected an image from the list in the dialogue box, press OK and drop the image where you want it in the terminal window.



As the images require a lot of memory, you should avoid using too many images and they should be kept as small as possible. A few smaller icons can be good to use for navigation help for the machine operator. As all images are saved in a shared table the image memory is not taken up by how many times a certain picture is used.

The total available memory space for images and utilised part is shown in the image selector dialogue. The utilised part is presented both as an absolute value and as a horizontal bar.

## Multiple languages

The terminal has support for handling different languages. Each text string specified is indexed and added in a table of text strings. Only one default language is used initially and is the language used unless the user selects another. It is the default language that is first displayed in the property fields in the program window under the terminal window.

In theory, you can add an unlimited number of languages, but at the time of print there is only support for the languages that use the same fonts which restricts international use somewhat. To add a language, you select Tools -> Translator... All the text strings are displayed that are defined in the translation table with one string per row. Each column corresponds to one language and you add a language by clicking Add. Enter a descriptive language name in the column header and then complete the translations in the relevant rows. By right-clicking on a row, you can select to jump to the terminal window, using the context menu, with the menu that the current text row is linked to in order to see the context the text string appears in. You can also hide the language columns that you are not interested in for the moment by right-clicking on the column header and selecting Hide in the context menu. In order to show a column again, right-click on the visible column and select the column header from the Show sub-menu.

## Terminal script

The terminal scripts are small programs that are linked to the terminal's function. A brief description of a terminal script is presented here as well as how you can use CANmaster PC-Tool to associate the script with menus and develop the terminal script.

For more information on terminal scripts and how you can develop them, see 'CANmaster Syntax reference guide' where there is a more detailed review of all the functions and how you write terminal scripts. For a description of the ready-to-use functions that have been produced, see the Terminal Script chapter in the 'CANmaster functions library'.

### Terminal scripts in general

The terminals scripts give you the option of making the terminal smarter. In addition to the ready-to-use terminal objects that you can place in the menu screen, you can create small programs that are run in the terminal. These programs or scripts can be used as follows:

- Converting measured data for presentation.

- Automatic setting of parameters and sensor calibration.
- Function testing of the system.
- Improving and developing more advanced user interfaces.

A terminal script that is linked to a menu has a number of variables and storage words that the script can use. When a menu that has a terminal script linked to it is activated, the terminal script is restored and is restarted from the beginning and the properties for all terminal objects in the current menu are restored to the values specified in the design phase.

A terminal script is developed using a special programming language at a low level. As it can take quite a time to learn to utilise the possibilities offered, there are a number of ready-to-use functions developed for you to choose from. You can get started by writing your own scripts by using the ready-to-use functions.

## Enhanced terminal functions

The most frequently used functions for enhanced functionality for the terminal using terminal scripts have already been developed for you to use. If you are missing any function, you have the option of producing your own functions or modifying the ready-to-use functions to better suit your needs.

When you open a terminal script it appears in the text editor where you can change the script in the way you prefer.

Create a new terminal script with the menu command File -> New -> New Script File -> Terminal Script File.

The terminal scripts are read and compiled as executable code when the terminal program is compiled. Any compilation errors caused by syntax errors are displayed in the message window. While you develop your terminal script, you can also right-click the script name in the project browser and select Compile from the context menu. If any error is identified in the script, you can double-click it to jump to the row in the terminal script where the error is.

## Terminal script and menus

Each menu can have a terminal script linked to it that is activated when the menu is activated. Specifying a terminal script for a certain type of menu links the script to the menu. Each time the menu is viewed in the terminal display, the script will start. When the terminal script starts, all the memory areas that belong to the script are cleared and all the terminal objects included are restored to their default settings.

To link a terminal script to a specific menu screen, see the section called "Menu structure and navigation". The presentation and update of parameter values are governed totally from the terminal script. These terminal scripts are automatically included in a menu where there is at least one parameter object included.

## Configuration parameters

Up to 48 16-bit configuration parameters can be set for the terminal. The configuration parameters are not set to zero when the other memory areas are set to zero nor when the system is restarted. Using these configuration parameters, it is possible to save the settings that remain and can be read again the next time you view the menu. All these parameters are shared by all terminal scripts, which is why it is important to consider the parameters that are used for each script. It is up to you to ensure that the parameters you read in your script are correct. You must also ensure that the various scripts do not use the same parameters for different things. This means that when you instantiate default functions, you will probably have to go in and modify certain values in the beginning of the files to avoid them colliding with each other.

Avoid updating these parameters more than is necessary. The parameters are saved in an EEPROM type memory, and the service life for the memory is depending on the number of updates. Entering new values in this memory several times per second will cause premature damage to the memory.

## Compiling terminal programs

In order to download the program file including all menus and functions you created for the terminal with the file extension `hydmd`, it must be converted to a compiled binary file. You create this file by pressing the Build button in the terminal window. The result is a compiled file that you name yourself but must have the file extension `hydmdh`. It is this file that is to be downloaded to the terminal when downloading the software to the control system.

To produce a compiled program file for the terminal, do the following:

1. Open the terminal program file with the file extension `hydmd` that you want to compile.
2. Click the Build button from the global settings.
3. Specify the file name for the resulting file. If you do not specify the extension `hydmdh`, this will be added automatically.
4. Click Save to create the binary file and close the dialogue box.

You can create several different program files for the terminal in the same project, for example for different machine versions, but you can only download one of these menu files to the terminal on the machine.

For further information, see the section called "Download of application program via Master/Terminal".

# Display programming

## Files

These are the types of files used.

Filename	Description
*.hydgui	The application. This contains the layouts and scripts for the application. Each application will have one single <code>hydgui</code> file.
*.hydobj	Object files. These are used to draw things on the screen. The <code>hydobj</code> files are generic - you can reuse them in various applications.
out.hydmh	compiled application. This is your application in compiled format. During compilation the <code>hydgui</code> and all the <code>hydobj</code> files are compiled and linked together into this file.

All the bargraphs/meters/input boxes/etc in the toolbox are `hydobj` files.

## Interface Overview

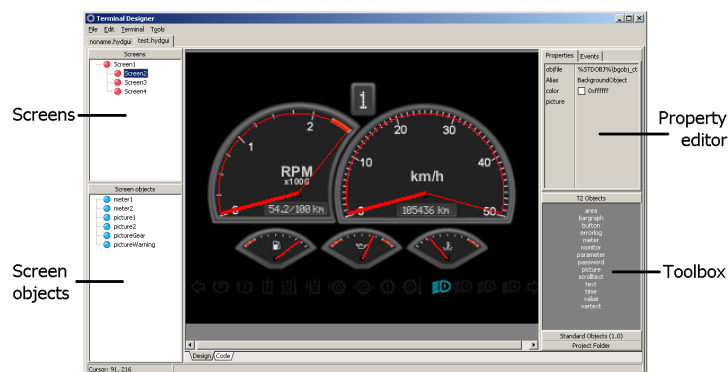


Figure 7.1. Screenshot

### Screens

This list contains all the screens in your application. You can have as many screens as you want.

### Screen objects

This list shows all objects on the current screen.

Property Editor

Here you configure the selected object. Under the `Properties` tab you configure the data. Under the `Events` tab you can configure the events.

Toolbox

These are the objects available that you can use to design your application.

## Designing an application

Add objects through the toolbox.

Configure objects on the screen through the property editor.

## Configuration of functionality

Basic functionality is configured through the designer.

### Keyboard input

All you need to do to get the basic functionality is to drag and drop the appropriate objects. The basic functionality will allow you to navigate on the screen, edit values, etc.

More specific actions are not handled automatically though - for these actions you must write script.

## Scripting

Fundamentals:

- Scripts are written with `Lua`. For more information about `Lua` see [www.lua.org](http://www.lua.org).
- The scripts are case insensitive.
- The application is event driven.
- Object properties are accessed as functions.

- To read the value of `bargraph1.value`:

```
x = bargraph1.value()
```

- To write the value of `bargraph1.value`:

```
bargraph1.value(x)
```

### Application scripts

To customize the behaviour of the application, scripting is often needed.

To write an event handler:

1. Select the object
2. Go to the `Events` tab in the property editor
3. Click on the event that you want to handle
4. The script editor is shown. Write the script.

## System API

`gotoscreen(screenalias)` Use this function if you want to go to another screen.

Example:

```
gotoscreen(screen2)
```

## Keyboard input

When you need to handle keyboard input in the application, you do this in the `keypress` event handler. To create this event handler:

1. Unselect all objects on the screen
2. Go to the `Events` tab in the property editor
3. Select the `Keypress` event

This function should return 0 if the key is unhandled and 1 if the key is handled.

Example:

```
function backgroundobject_keypress(k)
  if k == KEY_UP then
    -- handle the up key ..
    -- then return 1 to signal that UP is handled
    return 1
  end

  -- return 0 to signal that the key wasn't handled
  return 0
end
```





# Parameters

---

This chapter describes the function of the parameters and what they are used for, as well as the security aspects that apply to parameters. An account is then given of how CANmaster PC-Tool is used in the various phases of the development work for handling and configuring the system parameters.

## General

Parameters can be simply expressed as variable data values that are used together with standardised program segments to control machine functions with varying characteristics and limit values depending on the type of operation. Ramps and filters are examples of frequently occurring program functions where the parameters are normally included as an essential part of the construction of the program parts.

Using the same type of function block in different parts of the control program but with different parameters, one type of function block can be used to provide different computed results.

Examples of a parameter controlled function include the `RAMP` function, where the value for the minimum and maximum control current for solenoids is set with parameters, different types of filter and PWM outputs' characteristic with respect to ripple frequency and ripple amplitude.

All parameters included in the different parts of the system are saved in a common EEPROM memory. The memory is located in the Master unit and the information remains intact when the power supply to the control units is turned off.

There are three basic types of parameter; (1) function parameters that belong to the machine's application program, (2) unit specific parameters and (3) system parameters.

Each time the system starts, the parameters are transferred from the common EEPROM memory in the Master unit to the working memory in the various control units. The parameters that belong to the application program i.e. function parameters are transferred to the Master unit's working memory and the other parameters to the relevant control unit's working memory. When a change is made to a parameter value, both the EEPROM memory and the current value in the Master unit's working memory that applies to the function parameters are updated. The function parameters in the application program that can be changed via the terminal menu can therefore be changed during operation without the need of restarting the control system. The system must be restarted for the other parameters in order for the change to come into force. This applies for example to a change to the PWM outputs' ripple frequency and ripple amplitude.

If a change to a function parameter that is included in a terminal menu is not permissible during operation with the current machine functions modulated, but nevertheless should be easily available for adjustment, you can block this option with a program add-on in the form of a terminal script that is saved locally in the terminal's program memory. The operator must then stop the machine function, adjust the parameter and then test how the change has affected the program control.

The parameters are saved in a memory of fixed addresses that are computed by CANmaster PC-Tool when the control program is compiled. Each parameter is addressed using a block number and a record number. A structure of this kind is specified in the system for all functions that use parameters. These structures are numbered using fixed values as per the tables below. For parameters that are not system

specific but are connected to functions or to the application program, the block number is used for creating more instances of the parameter structure. The block number is a value between 0 and 255 that specifies the instance of the specified structure that is intended.

The table below describes the block numbers and record numbers that are used for the general configuration of the entire system and for user defined configurations in the application program that is not directly connected to individual functions.

*Table 8.1. General parameter block and record numbers.*

Block number	Record number	Contents
0 ... 255	1	Byte value parameters in the application program (EEPROM).
0 ... 255	2	Word value parameters in the application program (EEPROM).
0 ... 255	4	Long value parameters in the application program (EEPROM).
0x20 ... 0x3F	9	Byte value parameters in the application program (RTC RAM).
0x20 ... 0x3E	10	Word value parameters in the application program (RTC RAM).
0x20 ... 0x3C	12	Long value parameters in the application program (RTC RAM).
0 ... 255	96	Unit-specific parameters.
50 ... 65	97	Unit-specific data length parameters.
0	98	Standard language.
0	99	Name fields for J 1939.
0	100	Global settings for PWM.

The table below lists the record numbers that are directly connected to the parameters for each function.

*Table 8.2. Record numbers for standard functions with parameters.*

Record number	Function
101	PWMCH
102	FILTER
103	RAMP
104	ALARM
105	REGULATOR
106	BOOM
107	GEAR
108	RAMPDYN
109	PERCENT
110	AUTO
112	IOCONTROL
113	TRCONTROL

All block numbers between 0 and 255 can be used in the table above. By specifying -1 as block number, the block numbers are automatically assigned according to which are available for each type. This is used in the graphical programming editor. Once a project has been compiled, the assigned block numbers can be read.

## Unit specific parameters

In addition to the parameters that are connected to the various functions in the application program and the system parameters mentioned above, there is also the possibility of using unit-specific parameters.

These parameters are sent down from the Master unit to the control units when the control system starts and are used for the different types of settings of the base software in each control unit.

The unit-specific parameters follow the CANopen standard with an index and a sub index that specifies the parameter involved. In CANmaster PC-Tool these parameters are specified in text-based source code via parameters with record numbers 96 and 97.

Parameters with the type number 96 are used to keep the actual information that is to be sent to the various control units. The structure is defined as follows:

```
UNITREC DEFINE RECORD := 96
  Index:      WORD UNSIGNED
  SubIndex:   BYTE UNSIGNED
  Length:     BYTE UNSIGNED
  Data:       LONG UNSIGNED
UNITREC ENDDFINE
```

**Index** Index is the CANopen defined index value that the unit-specific parameter is to have. The value must be between  $2000_{16}$  and  $5FFF_{16}$ .

**Subindex** This information is also specified as per CANopen. The value must be between 0 and 255.

**Length** The length of the information that the parameter contains must be specified as 1 for 'byte-', 2 for 'word-' and 4 for 'long'-values.

**Data** The actual information to be sent to the control unit.

A parameter of this type has a CANopen parameter value that is retrieved by the control units via CANopen SDO procedures. If a unit uses several CANopen parameters they must all have sequential block numbers with no gaps.

To specify if a CANopen parameter is available for a control unit and if so how many, type 97 structures are used. The structure is defined as follows:

```
UNITRANGE DEFINE RECORD := 97
  RFrom:     BYTE UNSIGNED
  RTo:       BYTE UNSIGNED
UNITRANGE ENDDFINE
```

**RFrom** This parameter specifies the block number that is the first in the series of blocks used by a particular control unit. The value can be between 0 and 255.

**RTo** This parameter specifies the last block number in the series of block numbers that belong to the specified control unit. The value must be greater than or equal to RFrom and be no greater than 255.

The block numbers used for parameters of the type 97 are fixed numbers depending on the control unit that is intended. The following table specifies the block numbers that must be used for each control unit.

*Table 8.3. Unit specific parameter block numbers for CANmaster units.*

Unit	Block number
Master	51
Terminal	52
Joystick 0	54
Joystick 1	55
Joystick 2	56
Joystick 3	57
Expansion 0	58

Unit	Block number
Expansion 1	59
Expansion 2	60
Expansion 3	61
Crane 0	62
Crane 1	63
Crane 2	64
Crane 3	65

Normally, the standard values of the unit parameters should work fine but if any of the settings needs to be altered it can all be done from the projects settings dialog.

## Joystick parameters

The joysticks utilise the option of using unit-specific parameters to make settings that affect their function. This section describes these with examples of how their standard values can be adjusted. The adjustments are done from Project -> Properties where Settings is selected for the joystick to adjust.

Parameters in the dialog below are used to adjust the function of the base software that is used for the joystick. The measured signal value is filtered in the program to avoid small changes to the signal value, caused by vibrations, giving unstable control signals. The joystick's base software also handles the ramp values to counteract jerky joystick movements.

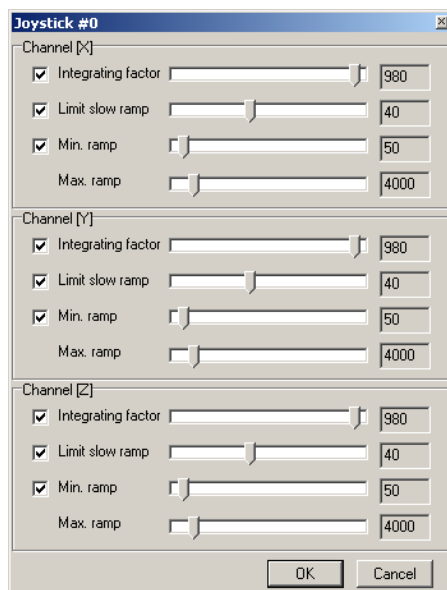


Figure 8.1. Joystick settings. Used to adjust the filter characteristics of a joystick

Filtering and damping of the signal values can be adjusted using the above joystick parameters.

The integrating factor determines the proportion of earlier output signal values that are to decide the output signal value when changes are made to the input signal's value. The integrating factor is specified as an integer between 500 and 999. A large value for the integrating factor gives a more stable output signal but a slower response to quick joystick movements. The value 980 is used as standard. To remove the integrating factor, uncheck the corresponding check box.

After the signal value has been filtered, the signal is damped in the form of a ramp. The size of the damping depends on the difference (dS) between measured signal value from the joystick and filtered

signal value. If the difference  $dS$  is less than the specified limit value, a linear ramp factor is calculated depending on  $dS$ . If the  $dS$  is greater than the specified limit value, the maximum ramp value is used, and if the difference is close to zero, the minimum ramp value is used. See figure below.

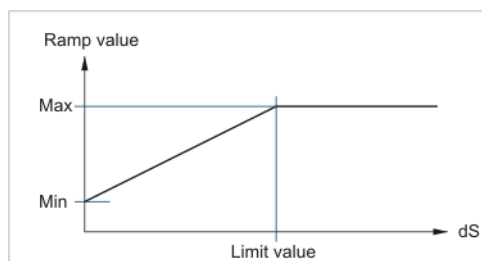


Figure 8.2. Calculation of the proportional ramp if  $dS$  is less than the specified limit value.

- The Limit slow ramp value is specified as an integer between 0 and 100. The value 40 is used as standard.
- The minimum ramp value is specified as an integer between 1 and 999. The value 50 is used as standard.
- The maximum ramp value is specified as a value between 1000 and 30000. The standard value is 4000.

The joystick software makes all the necessary computations for each axis X, Y and Z and then sends the results to the Master unit for modulation of the outputs to the valves. The computed values will be in a range 1 - 1023 which corresponds to each joystick axis range, -50 % to +50 % deflection.

If damping to the joystick's output signal value is not required, the limit can be set to 0 and the minimum ramp value can be set to 1. Only the specified value is then used for the maximum ramp that can then be set to an appropriate value to achieve the required response to the joystick. The greater the ramp value the quicker the control signal is allowed to change.

## Parameter editing

All parameters used in the system are created automatically by CANmaster PC-Tool. Most of these parameters are connected to the various standard functions that are used in the application program and some are system-wide or user-defined. All parameters that are included in the application program can be divided into categories according to how they will be used and handled.

### Operator parameters

These parameters are often the ramp values and suchlike that affect how the machine operator perceives the response from the machine. The parameters that affect this are usually available via the terminal so that the actual operator can adjust them for testing the best value.

All operator parameters must be added to the various menus in the terminal in order for the operator to be able to adjust them during operation. These parameters affect the response of the system from the input signals that are often affected directly or indirectly by the operator.

### Machine parameters

The machine parameters make up the various limit values for gear shifting, for example, or the maximum or minimum control current for the different proportional valves. These parameters should not normally be available for changing directly from the terminal when the machine function is modulated as an incorrectly specified value may jeopardise the safety of the machine.

These parameters can either be added under password protected menus in the terminal or be exclusively adjustable via a service

computer and only be adjustable when the machine function is stopped in order to increase safety when testing. The parameters that are to be made available for any kind of adjustment must be determined by the person who develops the programs for the machine. All parameters that are directly adjustable from the terminal must also be made available for adjustment from CANmaster PC-Tool.

**Safety critical parameters**

Other parameters that do not fit in any of the above categories should not be able to be changed by the machine operators or service engineers. Parameters in this category could be parameters for delay constants for a gear shifting sequence or calculation factors for regulators. An incorrect value for these parameters could result in serious safety risks for both damage to machinery and personal injury and should under no circumstances be changed by anyone other than the machine manufacturer's responsible system developer.

All parameters are always available to the system developer for adjustment in various places in CANmaster PC-Tool. The function specific parameters are always close at hand in the graphical programming editor by right-clicking on a function block and selecting Set Parameters. This menu option is only available for function blocks that include configuration parameters.

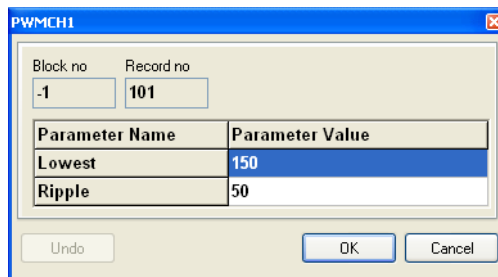


Figure 8.3. The dialogue box for the configuration parameters for a PWMCH function.

In addition, all parameters are listed in the project browser under Parameters. The browser shows a tree view with all the function names in the top level and under these the relevant parameter names with associated parameter value.

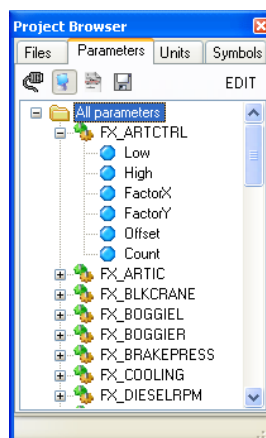


Figure 8.4. The parameters are shown in a tree view under each symbol name that is assigned to the function.

The division of the parameters into categories takes place in two stages. In the terminal program the parameters that can be changed via the terminal are specified directly. To be able to restrict the display

to certain menus, password protection can be added to differentiate between operator parameters and machine parameters. The parameter editor is used for division into categories in CANmaster PC-Tool.

Only the parameters that are added to the parameter editor are accessible for adjusting in the service version of CAN-master PC-Tool. In this way, the parameters are divided up into machine parameters and safety critical parameters.

All parameters used in the application program are saved in binary format in the file out.par when the project is compiled. The parameter file can be uploaded and downloaded between CANmaster PC-Tool and the Master unit in full, but it is also possible to update individual parameters directly during operation. For more information on uploading and downloading of the full parameter file, see the section called "Downloading various items of software".

When CANmaster PC-Tool is connected to a system in operation, the parameter editor can be used to adjust individual parameters. The parameter editor connects to the Master unit and synchronises the parameters in the editor with those that are in the Master Unit. Using the parameter editor lets both the system developer and the service engineer adjust the parameters that are added for editing in the editor. For other parameters, a developer version of CANmaster PC-Tool must be used, whereby parameters are changed by double-clicking them in the tree view.

To adjust safety critical parameters during operation, the system developer can double-click directly on a parameter in the tree view listing all the parameters. If CANmaster PC-Tool is connected to the Master unit you have the option of adjusting the parameter that is selected. This gives the option of changing all parameters in the system directly from CANmaster PC-Tool.

During the simulation of the application program in CANmaster PC-Tool, you can adjust the parameters directly in the graphical view by right-clicking a function block and selecting Set Parameters. The current values are then displayed for all inputs and outputs as well as all parameters. Clicking one of them will open a small window that allows the entry of a new value. A changed value does not have any affect until the configuration window that was opened is closed. This allows the possibility of making more changes simultaneously, and lets all changes be active at the same time. Note, however, that the changes that are made to the parameters during a simulation are not saved in the project following the completion of the simulation, which is why new values must be entered in the normal way after they have been tested in the simulator.

## Parameter Tree

The designer can create customized parameter editors much like the customized terminal views. A parameter editor allows changing parameter groups directly from the CANmaster PC-Tool in various ways. This is done in `Project Browser`, under the tab `Parameters`.

The parameter editors can be used to modify parameters in a file, a simulation or in a connected system. However, parameter editors can not modify parameters in source files.

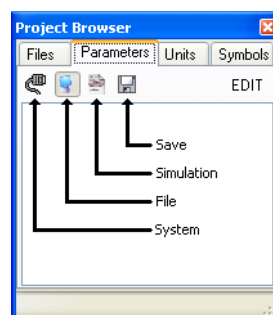


Figure 8.5. Parameter tree toolbar

A short description of each tool:



System	When you want to modify parameters in the system, click this.
File	When you want to modify parameters in a file, click this. By default the <code>out.par</code> file will be used.
Simulation	This is only available to users with designer access. During simulation this will be selected.
Save	When you have modified parameters through parameter editors, this tool is used to save your modifications. The parameters will be saved in the chosen target; System, File or Simulation.
EDIT	This is only available to users with designer access. When EDIT is pressed, you can modify the parameter tree and the parameter editors but you can't modify parameters.

Modifying the parameter tree is done from the context menu in the parameter tree view.

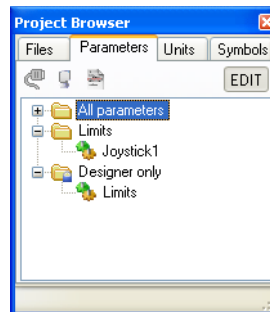


Figure 8.6. Parameter editor tree.

It's possible to restrict access to folders and editors. When right clicking a folder or editor in `EDIT` mode a context menu is shown. In this menu there is a submenu `Security`. There are currently 3 security choices; Everybody, Designer only and Password. With these choices you specify if everybody, only the designer or only those who know the password will have access.

## Note

You can never limit access for designers. Designers will always have access to everything in the parameter tree.

## Creating Parameter Editors

Parameter editors are dialog windows that allow users to modify parameters.

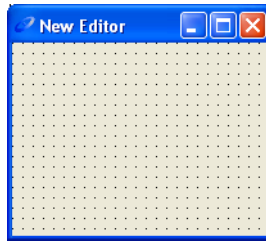
When creating parameter editors there are 2 things you must know:

- How to design the user interface
- How to define the calculations

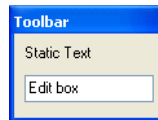
The fastest way to create a parameter editor for a parameter record is to right click the parameter record and select Auto create parameter editor.

To create a new empty parameter editor, goto the parameter tree described in the previous section. Make sure the `EDIT` button is pressed. Right click on the parameter view and select New Editor in the context menu.

To open a parameter editor, double-click on it in the parameter tree.



You can add various control through drag and drop. You can for instance drag a parameter from the All Parameters. The parameter editor toolbar also has tools that can be dragged.



Parameter editor calculations are similar to spreadsheet calculations (such as those in Excel). In a spreadsheet, cells can have formulas. In a parameter editor, fields can have formulas.

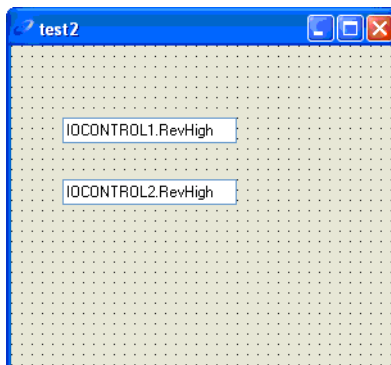
In a parameter editor there are 4 formulas:

- Init      Formula that initializes the field.
- Update    After the user has entered a value somewhere, the parameter editor is updated. Each field will be recalculated according its 'Update' formula.  
  
Note that a field that has a 'Update' formula will automatically be read only - the user can't change the value.
- Min      Minimum value.
- Max      Maximum value.

*Example 8.1. Parameters.*

This example demonstrates how to create a parameter editor that modifies parameters.

1. Open a blank parameter editor in EDIT mode.
2. Drag parameters from the Project Browser to the parameter editor. Parameters can be found under the All Parameters heading if you have any.



Through this parameter editor the user can modify both IOCONTROL1.RevHigh and IOCONTROL2.RevHigh.

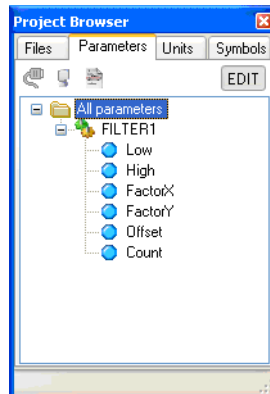
*Example 8.2. Calculations*

This example demonstrates how to create a parameter editor that performs calculations.

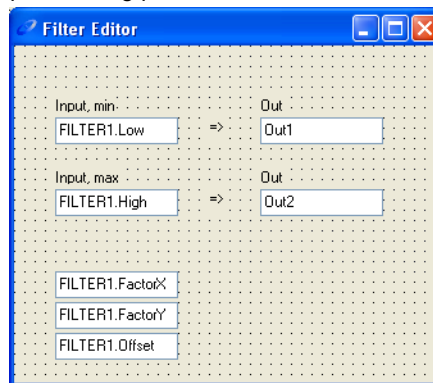
The goal is to modify a filter record. The parameter editor will let the user enter the minimum and maximum values on the input and also the corresponding output values. The parameter editor will modify the `Low`, `High`, `FactorX`, `FactorY` and `Offset` parameters.

Creating the GUI:

1. Create a project and add a FILTER function. Compile the project. Open the `Parameters` tab in the Project Browser.



2. Create a parameter editor. Open it. Drag parameters and controls onto it until it looks like this:



To add textlabels, use the "Static Text" tool in the toolbox.

To add `Out1` and `Out2`, use the "Edit box" tool in the toolbox.

To modify the text or name of a field on the parameter editor, double-click the field.

3. Double click on `Out1` and change the `Init` formula:

```
Filter1.Low * Filter1.FactorX / Filter1.FactorY + Filter1.Low
```

4. Double click on `Out2` and change the `Init` formula:

```
Filter1.High * Filter1.FactorX / Filter1.FactorY + Filter1.Low
```

5. Double click on `Filter1.FactorX` and change the `Update` formula:

```
Out2 - Out1
```

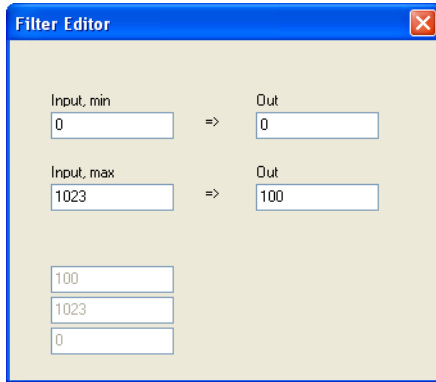
6. Double click on `Filter1.FactorY` and change the `Update` formula:

```
Filter1.High - Filter1.Low
```

7. Double click on `Filter1.Offset` and change the `Update` formula:

```
Out1 - Filter1.Low * (Out2 - Out1) / (Filter1.High - Filter1.Low)
```

Now the editor is completed. When executed it looks like this:



The Out1 and Out2 has the range 0 - 1000 by default.

The Filter1.Low should always be lower than Filter1.High. For Filter1.Low, Max could be:

```
Filter1.High - 1
```

If you wish to add this, simply goto EDIT mode and double click the Filter1.Low field. Then enter the formula in the Max box.

## Compare and merge parameters

A rather often reoccurring task is to compare two parameter files. For example, a factory default parameter file out.par as generated from the tool and an uploaded parameter file from a system where some adjustments have been made from the terminal. To help in this task the CANmaster PC-Tool has a special tool for showing the differences between two parameter files that also allows to merge changes between two parameter files. By selecting Tools -> Parameter Diff... an empty window as shown below will appear.

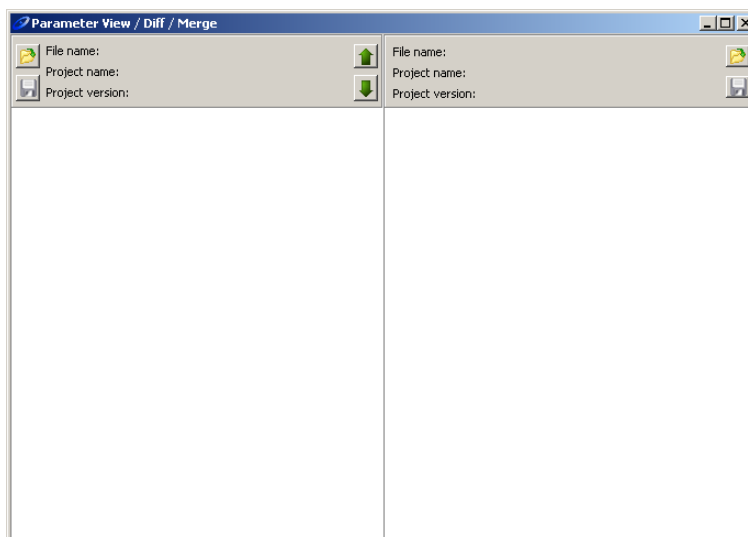


Figure 8.7. Parameter difference and merge window at startup.

This is the parameter comparison and merging tool. The window has a header information pane at the top that is splitted in two halves. The header information contain filename and project information from any loaded parameter file. Further on, the window is splitted in two halves, the left and the right panes. This

allows for showing two parameter files side-by-side. The parameter files are chosen by selecting the file browse button in each header part. When two parameter files have been chosen, one in each pane, the differences are highlighted to make it easy to see the differences between the files.

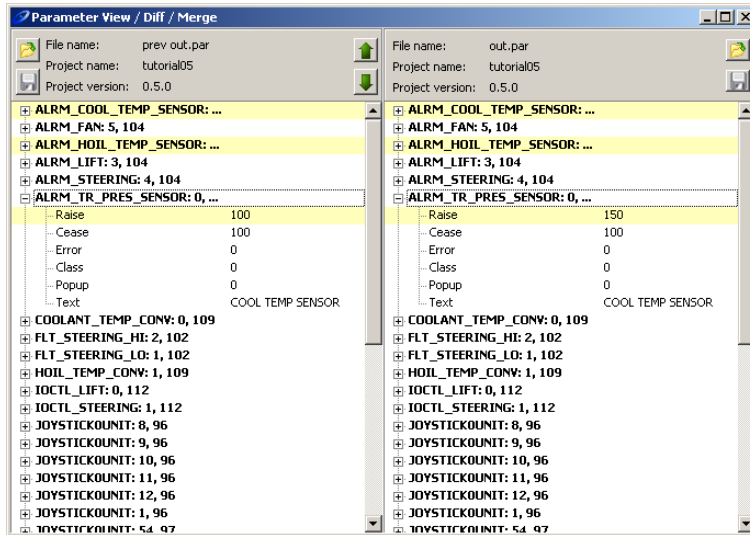


Figure 8.8. Parameter differences highlighted with two parameter files loaded.

When the parameter block is collapsed, the header row is highlighted but when the parameter data block is expanded the highlight is shown on each part that differs in parameter record. To quickly navigate between any differences, the two buttons in the middle of the header pane can be used to jump to the next or previous difference between the two files.

A changed value in one parameter file can be merged into the other parameter file by right clicking on the highlighted line and selecting Use parameter value of other file from the context menu. This will update the value where clicked, and the highlight will be removed, since the values are now equal. To save the changes to disk, the save button in the files header pane must be clicked.

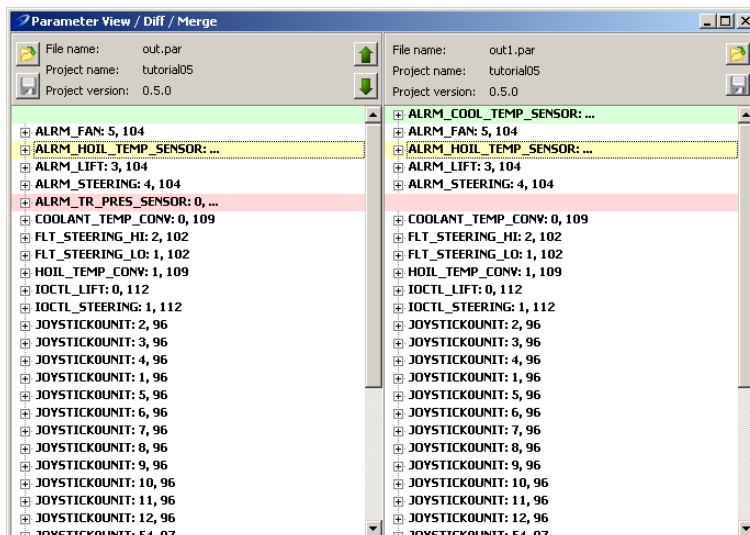


Figure 8.9. Missing parameter records are highlighted in red while added parameter records are shown with a green background.

In case the parameter files do not have the exact same parameter configurations missing or added parameter records are highlighted with different colors. The left side is used as the original that is

compared against. This makes it a missing parameter, highlighted with red, if the same parameter record is not available in the right hand side file. On the other hand, if a parameter record is present in the right hand side but missing in the left hand side, the parameter record line is highlighted in green.



## Compiling projects

---

This chapter describes the procedure for compiling program files in the project that are to be downloaded to the control system's Master Unit. The program files affected by this process are the application program, associated parameter files and general information on the project that is also saved in the compiled files. However, the terminal program that also belong to the project is compiled separately and are described in Chapter 6, *Terminal Design*. As the terminal program is dependent on the application program, it must also be recompiled when any form of change is made to the application program that affects this. See the 'Settings' sections in this chapter for more information on how this part can be automated.

The entire building process and compilation is done automatically when you activate the Build function in CANmaster PC-Tool.

The end results of the building process are compiled program files that can be downloaded to the Master unit. Compilation is only carried out if the program files do not contain any programming errors. If errors occur, the compilation terminates and a list of errors is presented.

The same compiled program files that can be downloaded to the control units are also used when testing the program in the CANmaster simulator. A project is often compiled several times during the development work to enable testing of the various parts of the program in the simulator before the program is fully complete and ready to be downloaded to the machine's control units. The 'Testing project' chapter describes how the control program is tested using the CANmaster PC-Tool simulator.

When all testing at a desktop level is complete and the program seems to work as intended, you download the program to the control units for testing with the machine in operation.

### Compiling application program

In order to build and compile a project, you select Project -> Build or click the Build button in the toolbar. The process of assembling all program files for compilation begins and a log window with messages shows the progress of the process. The log window shows the files that are included in the compilation, any error messages and warnings and, if everything goes well, how much of the different types of memory was used. An example of how this information might appear is shown below:

```
Files to build:
transmission.src
geara.src
vipp.src
turn.src
...
IRAM:   FD00-FD7F, used number of bits = 68
XRAM:   C080-C7FF, used number of bytes = 307
Flash:  used number of bytes = 15064
EEPROM: used number of bytes = 1021
Errors = 0, warnings = 0
```



If the compilation went well, without any errors or warnings, the compiled files are created for downloading to the Master Unit. The same files are also used to simulate the control program in the built-in simulator.

## Correcting compilation errors

What is often most interesting to check is the amount of memory of different types that is used, and of course if any errors or warnings have been detected. If the numbers after Errors or Warnings on the last row are not zero, there is some form of error in the source code that must be corrected. All error or warning messages are shown before the information that shows the used memory.

### Errors in source code files

Depending on the type of source code file that contains the error, CANmaster PC-Tool will perform different operations:

If it is a text-based source code file, the file containing the error will be opened and the offending row will be marked. All identified errors are listed in the log window, but only the first error encountered will be marked in the text editor. Double-clicking an error in the list in the log window will open the file containing the error and the row where the error was identified is marked. If you single-click on the error message instead and press **F1**, more detailed information will be displayed on what may have caused the error along with suggestions on how to correct the problem.

If an error is identified in a graphic program file, this is also opened and the part of the relevant function block that is incorrect is marked in red. The first file that contains the error is also opened in this case. However, all errors are marked directly in the graphical program window. Moving the cursor over an error displays an explanatory text of what is wrong.

### Utilized memory

Information on the proportion of the storage areas in the Master Unit that are utilised by the control program are reported at the end of the message providing the compilation was successful. Different types of data use different storage areas, which is explained in brief below:

IRAM shows the used memory for bit variables. Available memory corresponds to 1024 bit values. Apart from the number of bit variables, the memory addresses that are used to save these values are displayed.

XRAM specifies the used number of bytes for other variables that are not bit values. In total there is the possibility of using over 1900 bytes for these variables.

The information appearing after Flash specifies the proportion of memory that is used by the executable program code. In total, a control program can use up to 128 kB which corresponds to 131072 bytes.

Parameters used by the standard functions and any user defined parameters are saved in an EEPROM memory. Data stored in this memory remains when the powersupply for the system is turned off. In total there is space for 64 kB. The available memory space is shared by alarm and the central error log, which is why all memory cannot be used for parameters. This is not usually a problem, as the amount of memory that is normally used for parameters seldom exceeds 2 kB.

### Note

It is not just the use of the system's available memory that must be checked, but also the use of processor time and the load on the CAN bus must be reasonable enough for the system to be able to work as intended.

The used CPU time and load on the CAN bus is measured automatically when simulating the control program in CANmaster PC-Tool. See Chapter 10, *Testing projects* for more information.

## Compiler settings

There are a number of settings that affect the results of the compilation process. These settings are made under the General tab in the dialogue box for project properties. You can access this dialogue box via Project -> Properties.

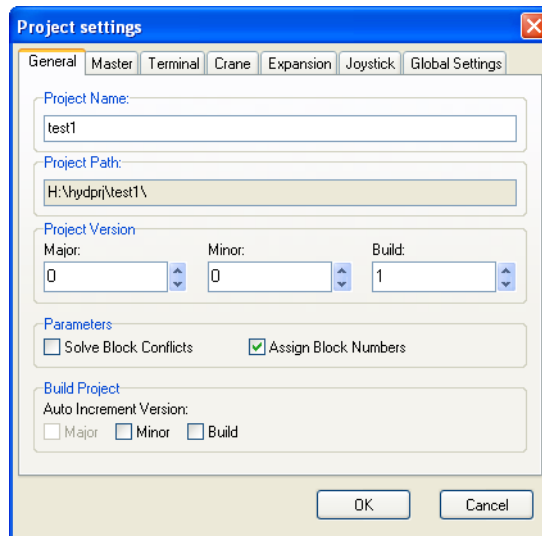


Figure 9.1. The dialogue box for general project settings that affect the compilation process.

The dialogue box is divided into three main parts. The project name to be added to the system software is shown first and then the current version number. The project name together with the version is used in the system to check that the correct software is downloaded to all units. This information only affects the compilation to the extent that the data specified here is saved in the compiled files.

The bottom part of the dialogue box affects the settings that control the compilation process and handling of the above mentioned information. By marking one or more of the check boxes by the version number, the corresponding figure in the version number will be automatically incremented one step each time compilation is executed without any errors. In this way it is possible to get a certain amount of automation in the compilation process for handling the version number in the system software. A more detailed description of the automatic incrementing of the version number you will find in section the section called "Version number".

The last setting that can be made when recompiling the application program is to check if the program for the terminal needs to be recompiled. As the terminal program depend on data and symbol names that are present in the application program, there is a dependency between the application program and the terminal program that means that the program for the terminal must often be recompiled when changes have been made to the application program. By marking the Yes check box, you ensure that this is done automatically when the application program is recompiled. For more information on how you compile the terminal program manually, see Chapter 6, *Terminal Design*.

## Downloadable files

When compilation is complete without any errors being reported, a number of files are created that together make up the program that can be downloaded to the control units. The same files are also used by the simulator to test run the control program in CANmaster PC-Tool. The files that are created include the actual application program, the parameter values and general information on the complete control program.

- out.hex     The application program is in `out.hex` and is the executable code that contains all control logics. This file is downloaded to the Master unit and is referred to in the downloading process as Application.
- out.hdr     The application specific information that contains the information required by both the terminal and the Master unit and for service and troubleshooting is in the `out.hdr` file. This file is also downloaded to the Master unit and is referred to as Header.
- out.par     The parameter values are saved in the file `out.par` and can be downloaded if required. The parameter file must be downloaded if there are new parameters or if the parameters

values have changed. For more information on the handling of parameter downloading, see Chapter 8, *Parameters*. Note that the old parameter values are overwritten when a new parameter file is downloaded to the Master unit.

For the separate compilation of the program for the terminal, the following file is generated that can be downloaded to the control system:

out.hydmh    The terminal application holding the menu designs and terminal scripts in binary form. This file is downloaded to the Terminal unit and is referred to in the downloading process as Application.

# Testing projects

---

A very important part of the development work is testing. CANmaster PC-Tool includes a number of advanced tools for handling these parts of the project. You have access to both text based and graphic aids for testing your program in various ways. You also have access to advanced graphical and text based functions for documentation of test results, whereby the data files created can also be used for analysing test results using external analysis programs.

The same functions that are available for testing and logging during program development are also available when CANmaster PC-Tool is used for service work out on the machine.

The built-in simulator in CANmaster PC-Tool uses the same compiled program files as those downloaded to the control units. You thus simulate the control program under real conditions as if it were downloaded into the control units. You can select to run the program at full speed or to step through the entire program or parts of it to pre-set breakpoints for closer analysis. So as to be able to see how various program sections work when the input signals vary over time in a specific order you can either manually control the input signals or create a script which automatically simulates input signals and shows output signals in real time in graphic form.

In this chapter we go over how best to utilise CANmaster PC-Tool for testing and troubleshooting and which possibilities are available.

## Simulate projects

This section describes how CANmaster PC-Tool can be used to test the control program in a simulator environment before it is downloaded into the machine's control units. Because it is the compiled files that are used in the simulator - the same files that can be downloaded to the control units - you must first compile the program and rectify any errors reported during compilation.

### Simulation of control logic

The most important consideration when testing a program is to first check that the basic logic functions are as intended, i.e. that program functions are activated in the order and under the conditions controlled by the logic terms in the program. To check this there are a number of aids built into the simulator.

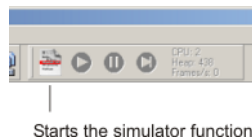
For example, you have the option of stepping through the program line by line and reading off I/O status or running the program in real time. There are also possibilities of setting breakpoints in the program where execution is stopped in order to check special parts of a course of events.

The part you will probably use most is the various graphical aids available in CANmaster PC-Tool to see what happens when you run your control program in the simulator. For example there is the option of logging signals over time in the same way as the logging is performed on a machine in operation to get an idea of what happens in the event of complex courses of events involving many input and output signals activated simultaneously. In order to automate the testing you can create test script that controls the input signals in order to as far as possible simulate a real course of events and test various operational cases.

## Open the simulator function

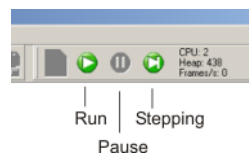
To start a simulation of your control program, all design windows and source-code files must be saved and the program compiled. The compiled files needed to run the simulator are uploaded automatically to the simulator when you start the simulator. If you have changed anything in your control program you will be asked whether you wish to re-compile the project before continuing. If you answer yes, compilation starts automatically.

The simulator's tool field is to be found in the main window at the top right where the buttons are shaded in grey when the simulator is switched off. You can open the simulator function by pressing the Open simulator window button.



Whilst the simulator is in operation, all design windows change to simulator mode and the possibilities of changing the program or the source code are then blocked. In order to be able to make changes in source-code files the simulation must first be concluded by closing the simulator window. The program must then be recompiled if any changes have been made.

You can start the simulator by pressing Run.



The various buttons in the simulator's toolbar have the following functions:

- |          |  |
|----------|--|
| Run      | Starts the simulation in real time. The control program is run in CANmaster PC-Tool at the same speed as when the program is downloaded into the control units. Whilst the program is in operation, input signals can be changed and output signals followed. The simulated control program's utilisation of memory, processor time and CAN bus load are also measured and displayed in real time. |
| Pause    | Pressing this button pauses the program. After having been paused it can then be restarted or stepped further.   |
| Stepping | To step the control program line by line of function block by function block instead of running in real time this button is used. There are a few different stepping alternatives available that are described below.  |

The simulator's tool field on the right also displays important statistics for the execution in the form of processor time (CPU), memory use (Heap) and the load on the CAN bus (Frames/s):

- |          |  |
|----------|--|
| CPU      | 'CPU' indicates the processor time as a percentage and should not exceed 50% for the program to work as intended. If the processor time exceeds 50% there is a risk of the regulating functions in the program not working as intended, which can cause a jerky or unstable system or slow reactions to various signals.   |
| Heap     | 'Heap' indicates the memory use during execution and should be below 1500 bytes for the memory areas not to be overloaded. If in any instance the value exceeds 1500 this may cause an unstable system.  |
| Frames/s | 'Frames/s' indicates the load on the CAN bus. The simulator counts the number of CAN-bus packages a second sent between the control units and the reading indicates how big a bandwidth of the CAN bus's available capacity is being utilised. The reading should be under 1500 packages/second so as not to cause collisions and congestion on the data bus, which can result in delayed signals or an unstable system. |

In the Simulation menu at the left of the main window there is a menu-selection list in which various parts of the simulation can be selected and set, e.g.: show I/O list (Display I/O), simulation of terminal function (Simulate Terminal), change I/O name between internal designations and symbol names (Toggle I/O name) and monitor selected signals (Trace & Watch).

## Simulation methods

Quickly finding problems and direct errors in the control program necessitates methodical work. In general this can be divided up into a few steps that you should follow in order to get as much as possible out of the simulator in CANmaster PC-Tool:

1. Try to localise the area of the program where you suspect that the error is arising.
2. Set a breakpoint at the beginning of the area, the diagram or the source-code file.
3. Run the simulation to the breakpoint and then step line by line and carefully follow the values of input and output signals as well as internal variables.
4. Identify the cause of the problem, change and restart until all errors have been rectified.

### Setting break points

A breakpoint is a place in the program that means the simulation/program run stops where you have placed this breakpoint. Because a control program for a complete machine can include a very large number of logical conditions and calculations it is not particularly efficient to step through the complete program line by line. With the aid of breakpoints you can instead let the program run continue unhindered until the breakpoint is reached in order to investigate I/O status in this very mode and then step the program run within a specific area or continue the program run to the next breakpoint.

To set a breakpoint in a graphical design:

1. Activate the design window containing the logic where you wish to set the breakpoint.
2. Right-click on the function block where the breakpoint is to be placed.
3. Select Breakpoint Item. A breakpoint is then activated on the function block in question.

To delete a breakpoint, right-click and again select Breakpoint item. If you have several breakpoints in the same design window you can delete them all at once by right-clicking on the design area and selecting Delete Breakpoints.

On a function block with a breakpoint the function block's name is displayed on a red background. When you run the simulation, the simulator stops just before the function block is executed. You can then look at the I/O status and see whether the current values seem reasonable. When a breakpoint has been set the text Break is also displayed in the left-hand margin in the source-code window at the relevant place in the source code. You can have up to ten breakpoints activated simultaneously in various parts of the control program.

To set a breakpoint in the source code file:

1. Find the line of source code on which you wish to set a breakpoint.
2. Right-click on the line of source code to call up a context menu.
3. Select Toggle Breakpoint.

To delete the breakpoint right-click on the line and select Toggle breakpoint.

### Stepping of the program run

To facilitate troubleshooting and simulation there are alternative ways of running the control program in the simulator. One alternative is to run the program in real time in the same way as if it was downloaded

into the control units. In this mode you have the option of controlling input signals and directly see how they affect the output signals. If you have detected a problem in any part of the program for a specific combination of input signals it may be appropriate to run the program more slowly in order to more easily see what is happening. You can then select to step the program in various ways.

There are three ways of stepping through a control program. You can select this by selecting Step Mode from the menu-selection list under Simulation.

The various options are as follows:

Program	Using this option the whole program performs one execution cycle, i.e. all program instructions are run once and then stop at the beginning of the next execution cycle. You can use this method to quickly step through the entire program without needing to set a breakpoint at any special place in the program code.
Source line	This option is the most useful stepping method, and the default setting if you do not select anything else. With this stepping method the execution point is moved one line in the source code every time you step forwards. A line of code is in this instance the code which you have written yourself or which has been generated from the program you have designed graphically. Most function blocks in the graphical editor correspond to one or more lines of source code.
Single instruction	Finally there is an even more detailed method of stepping, though it is seldom used. With this method the control program is stepped forwards in assembler instructions. This method is less interesting for troubleshooting of complete programs, and it is above all used for designing new function blocks.

At the same time as you step in the code, the equivalent place is shown in the source code. The design window displays the active function block with a green frame, and in text based source code the active line is highlighted. By stepping through programs that have been created using graphical design, you will now clearly see the order in which the various function blocks are executed.

## Displaying variable values

During the time you are running the simulator you can continuously check input and output signals with the aid of the signal monitors. For further information how to use them, see the section 'Monitoring signals' in this chapter. It may also be of interest to see values of internal variables and outputs on function blocks that do not directly control output values. Displaying values of them is very easy, and can be performed in a number of different ways. In the graphical programming window you have the option of seeing names and values of all memory references that the function blocks use or is updating on the occasion in question. If you left-click and hold down the left mouse button on a function block, all this information is listed in a small temporary window. In the simulator's source code window you can also double-click on a line, and the relevant window will then display the information on all references to be found on that particular line. If you right-click on a line of source code in the simulator window and select 'Edit status', a window will be displayed in which the value of the variable in question will be displayed.

## Trace and watch

To follow the value of a selection of variables over time you can put them in a trace list. The trace list is activated by selecting Trace and Watch from the menu Simulation.

The dialogue box displayed contains a list of all symbols in the program. Highlight the symbols for which you wish to follow the value in the left-hand window and click on the arrow button Add to transfer them to the right-hand window containing the active trace list. Symbols can be deleted from the active list by highlighting and clicking on the arrow button Delete. You can also double-click on a symbol in a window to transfer it to the other window.

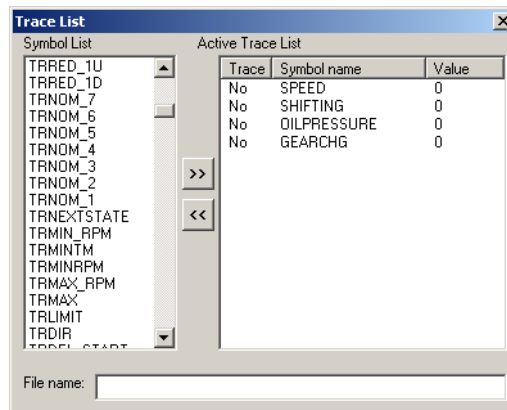


Figure 10.1. Example of a trace list in which all variables in the control program are listed on the left, plus the selection made in the right-hand section together with current values under 'Value'.

For the symbols you have selected to follow, the value can be saved as a function of time in a separate file for further analyzes. This is indicated by right-clicking on a symbol and selecting 'Toggle tracing' in the quick menu displayed. If no Trace file has been created, a dialogue box is first displayed in which you must name a trace file. Then right-click on the symbols you wish to add by selecting 'Toggle tracing'. Values of the symbols marked with YES in the column 'Trace' are saved in the Trace file.

### Display I/O window

In order to manually simulate the input signals to the control program with the values and in the logical order desired, it is possible to regulate them in the form of a sliding control for analogue signals and on/off buttons for digital signals. This option is very useful for initially checking that the logic conditions in the program are correctly designed for a specific function or combination of functions. This particularly applies to functions with a high security level for which under no circumstance may it be possible for a function to be activated if the input conditions do not follow a given sequence.

This simulation window is opened by selecting Display I/O from the menu Simulation. To display the selected symbol names instead of the internal variable names, select Toggle I/O name in the menu selection site under the menu Simulation.

All signals forming part of the program are displayed in the window with their name and current value. The value of an analogue signal is adjusted by dragging the sliding control with the cursor, whereby you can simultaneously read off the value. You can also continuously adjust the value by keeping the left mouse button depressed with the cursor in the slit to the left or right of the control handle, whereby the value continuously increases or decreases at a slow pace. It is also possible to directly enter a value. A digital input has a corresponding button that can be depressed (activated) or released (inactive). The status of the digital inputs is directly displayed on the button in the form of a 1 or a 0, whereby 1 stands for activated high signal.



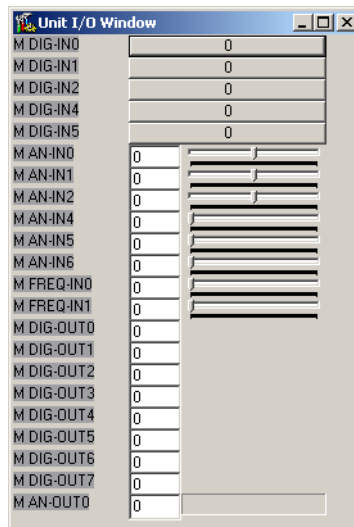


Figure 10.2. Example of manual control of input signals. The signals that can be controlled are represented by various types of controls. On the left is the signal's name. You can alternate between symbol name and internal name by activating the function Toggle I/O names in the menu Simulation.

The input conditions can in many cases be difficult or almost impossible to simulate manually in that with this function you can only adjust one input signal at a time where the signals in the real machine system come from sensors of various kinds that the operator does not directly influence. The logic conditions may also be dependent on the time between different input signals. In this case it may be appropriate to create a test script which automatically activates on/off signals and analogues signals in accordance with a specific pattern. There is a brief description below of how to write and use test scripts.

## Simulator scripts in general

A simulator script is a text file with commands that CANmaster PC-Tool performs during the simulation. By creating such a script you can automate large parts of the otherwise time-consuming and in many instances impossible task of controlling several parallel input signals in the desired manner during testing. For example, a script can change the reading of a diesel motor's revolutions when the target value is changed to simulate a diesel engine in operation.

### Creating simulator scripts

A simulator script is created by selecting File -> New script file -> Simulation Script File. Simulator script files have the extension `.hydsim`.

Simulator scripts are written in the programming language Lua, to learn more about Lua programming see <http://www.lua.org/pil/>.

A simple script that initializes a few inputs to 512 is as follows:

```
sym.JoystickX = 512
sym.JoystickY = 512
sym.JoystickZ = 512
```

This script will just set the 3 symbols and then exit.

When you want to create a script that manipulate an input signal over time, you want it to be executed continuously and not just once. If you create a function called `tick` it will be executed before every time step of the simulator. A simple script that emulates an engine is shown below:

```
function tick()
```

```

-- read the application symbol "accpedal"
requested_rpm = sym.accpedal * 2000 + 750

-- set value of the application symbol "rpm"
sym.rpm = requested_rpm
end

```

### Activating test script in the simulator

In order to load and run a script file in the simulator, the simulator must be stopped. The script file is loaded and run by selecting Simulation -> Load Script. When you then start the simulation the script will be executed.

## Monitoring CAN-bus signals

It is also possible to simulate CAN-bus packages on the secondary CAN bus (CAN-2). This can be used to test control logic for diesel engines that are controlled via a CAN bus in accordance with SAEJ1939 or other types of systems. Checking of these signals takes place with the aid of the CAN Monitor, which you will find under the menu Simulation.

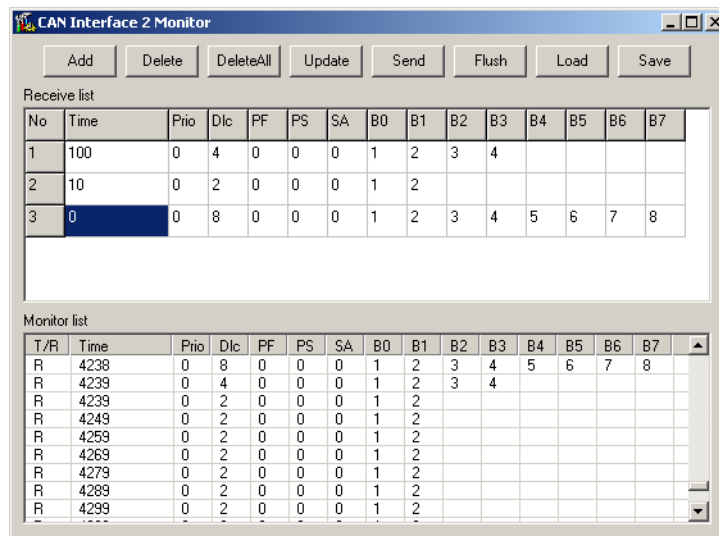


Figure 10.3. In the CAN monitor you can see what is happening on the external CAN bus. In the upper list you should indicate which messages are to be sent, whilst all dispatched and received messages are displayed in the lower list.

The control window for CAN monitoring is divided into three parts. From the top down they are as follows:

1. The command field, comprising a line of buttons used to control, check and handle the CAN signals that are simulated.
2. The recipient list is a list of all signals that can be checked. The list constitutes the data packages sent in to the Master unit, and the name thus comes from the Master unit, even if it is the packages sent from other units that are listed here.
3. The monitoring list includes all data packages sent and received by the Master unit on its external CAN bus interface.

With the aid of the CAN monitor you can simulate a diesel engine that is controlled via CAN-bus SAE J1939 and see how the control program sends information to the engine and how the program reacts to various values. The information that is set and monitored is at the lowest level, and all data packages received or sent must be indicated byte for byte.

## Simulating CAN signals

To simulate signals sent from a connected piece of equipment, the various messages in the upper list are defined. Every line in the list constitutes the definition of a message. Every definition is numbered, and this is displayed in the first column on the far left. This column is only used for seeing how many messages have been defined.

Other columns have the following functions:

Time	The stipulated time in this column constitutes the frequency at which the message is to be sent. The time is stipulated in the number of ticks of 10 ms each. A One thus indicates that the message must be sent to the master unit every ten milliseconds. If you stipulate a zero in this column, the message is only sent out by manually highlighting the line and clicking on the Send button in the command line above the list.
Prio	Stipulates the priority of the message.
Dlc	Stipulates the data length of the message. This value must be between 0 and 8, and stipulates the number of bytes in the data section of the message.
PF	Stipulates package format.
PS	Stipulate specific package format.
SA	The source address of the unit sending the message. This address is used to identify who has sent the message, since there may be several units who can send messages on the same CAN bus.
B0 - B7	Every column indicates the value of the corresponding data bytes in the package. The number of columns filled in must correspond to the value stipulated in the column Dlc, and values must be filled in from left to right. A non-stipulated data field is sent as the value zero.

To add several messages to the recipient list, click on Add. A further line which you can fill in is then added to the list. Dispatch of a defined message to the simulated Master unit does not commence until you click on Update.

To delete a defined message, highlight the line in question in the recipient list and select Delete. You can also delete all definitions by clicking on Delete All.

In the event that a message is to be sent manually, when the frequency time is set to zero, highlight the requisite line and then click on Send. A line is highlighted by clicking on the figure in the column on the far left or in one of the other cells in the requisite line.

To save the defined messages to a file for later re-use, click on Save. A file-selection dialogue opens, permitting you to select the name of the file to which the information can be saved. The ending of the file name will be `.can`. In a corresponding way you can open previously saved files by clicking on Open. All defined messages in the recipient list are deleted before a new file is inputted.

### Note

When you save the CAN monitoring list, only the message definitions are saved and not the monitoring list of all messages that have been sent and received.

## Simulation of terminal program

In the same way as you can simulate the application program in CANmaster PC-Tool, you can also simulate the terminal program you have created. You thus have the possibility of testing that the terminal and all the menus are functioning as intended. Because the terminal and the Master unit collaborate you must also simulate the application program in parallel to see the terminal's various functions.

To simulate the terminal program, do as follows:

1. Press the button 'Open simulator window' to start the simulation function.

2. Select Load Terminal SW under the menu Simulation to input the compiled terminal program.
3. Select the file you wish to simulate. Terminal files that are compiled have the file ending `.hydmlh`.
4. Select Simulate Terminal under the menu Simulation to tell the simulator that you wish to simulate the terminal in parallel with the normal control program.
5. Start the simulation by pressing the RUN button in the simulator's tool field. A window opens, displaying a depiction of the terminal's front panel, in which all buttons and the display work exactly like a real terminal. Use the mouse cursor to press the terminal's buttons.

## Monitoring signals

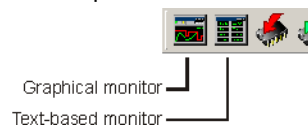
One of the most important parts of testing is being able to monitor and follow inputs' and outputs' values as functions of time. CANmaster PC-Tool includes two types of monitor: a text based monitor in which momentary values for selected I/Os are presented in tabular form and a graphical monitor in which I/O values are displayed as functions of time in the form of graphs in a diagram. The monitors can be used both for simulations of the control program and measurements in a control system in operation, with the same functionality in both test cases.

The tools for handling the monitoring are the same, regardless of whether you wish to look at signals from the simulator or from a connected control system on a machine. When you start the monitoring, the program recognises whether or not the simulator is in operation. If the simulator is not activated, CANmaster PC-Tool automatically tries to connect to the external control system via the RS232 port and upload measured data.

### Note

To get the monitor in operation with the simulator running you have to open the simulator function before you open the monitor. Otherwise the situation remains where the PC-Tool program tries to connect to the RS232 port for monitoring of the signals from a connected external control unit.

The buttons for starting the type of monitor in question are to be found on the tool field in the main window:



### Text based monitor

To get an on-the-spot picture of current values in the running control program you can use the text-based monitor in which values are presented in tabular form for selection of the I/O you require. So as not to unnecessarily load the system, all symbols are deactivated from the start, which is evident from the fact that the readings either include a small dash or are displayed in grey. When a value is active it is displayed in green. By clicking on the symbol's name you can activate and deactivate it.

This function does not offer any options of saving values for later analysis, but it provides a good picture of how the control program works and which sensor values are being obtained. The monitor can be constantly connected in order to catch particularly interesting parts of the program control. The more active symbols you use the more time is needed for every value update. To reduce the load, no more than five symbols should be active at a time. A monitored value is uploaded from the system every 50 ms. The more values that are followed the more time passes between every update of a specific value. In the event of rapid courses of events you may thus need to monitor for a longer time in order to catch interesting parts.

### Graphical monitor

The graphical monitor has the great advantage that you can follow signal values as a function of time in the form of a graph. To start monitoring, press the button Open Graphical Monitor on the toolbar. An empty diagram is displayed, with the time scale on the X axis and the value scale on the Y axis.

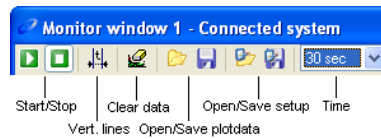


Figure 10.4. The graphical monitor's toolbar.

The measurements begin when you click on the green arrow button on the far left of the tool field. To terminate measurements click on the stop button. When measurement has stopped, no further readings are saved. If you then select to start the measurements again, there will thus be a jump to the new values. In this case the diagram displays as a straight line the time that has passed without the readings having been updated.

## Operation

Every diagram you use can have a number of different settings, which means you can display the very information you are interested in. Below there is a description of some common settings for the graphical monitor and the way the diagram area can be adjusted by zooming and panning along both axes.

## Settings

A basic function of this type of diagram is the time resolution. In order to allow easy selection between different time resolutions there are a number of predefined time scales. They are displayed at the top right of the diagram window's tool field. With the aid of the predefined scales you can quickly change the time scale to be used. The current time scale is only used for displaying data, and does not affect the sampling frequency of measurements. The readings are always sampled at the maximum possible frequency.

## Adding and configuring signals

The measurement channels are displayed in the diagram in separate colours. To add a signal to the measurement, right-click in the diagram area and select Add signal. In the submenu then displayed, all the control program's symbols are available, grouped in accordance with the type of value they contain. For example, analogue inputs are displayed in the submenu Analog In, whilst bit values (on/off-signals) are to be found under Bit Memory, etc.

## Tip

For PWM signals you can measure the control current value by adding the analogue input (the common return input) which belongs to a double-acting PWM output.

The signal's value can also be adjusted to make it easier to read in relation to other signals in the diagram. To set the scaling for a specific channel, right-click on the diagram area and select the channel under Edit.

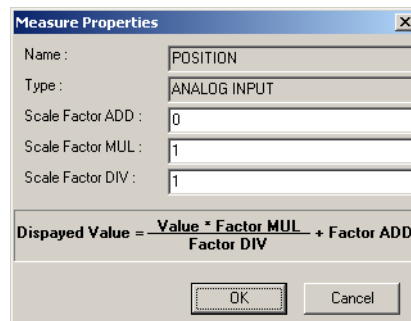


Figure 10.5. The dialogue box allows you to rescale every single channel in accordance with a given formula.

In the dialogue box you will see from which symbol the channel is measuring values and the type of signal. Under it there follow three text fields that you can use to rescale and modify the result displayed.

At the bottom of the dialogue box, above the buttons, there is also a display of the formula used for the value displayed.

This function can be useful if you wish to separate two analogue signals so that they are displayed more clearly, separated from each other in the diagram window. You can adjust this in the field Scale Factor ADD for the two signals so that one has a positive value and the other a negative value.

## Note

The changes you make to the channel-specific settings are not implemented until you click on OK.

The number of signals monitored determines the update speed for every value. Because of the restricted bandwidth in the communication between CANmaster PC Tool and the connected control system, only one variable per 50 ms can be measured. The more signals that are monitored, the longer will be the time between updates for every value.

During simulation, the same behaviour will normally be used, but it is possible to get faster measurements. To increase the speed, click on the menu Settings and then Tool settings. In the Measurements settings, change the measurement speed to `High`.

## Scrolling and zooming

You can scroll by dragging with the mouse on the timeline at the bottom, the Y-axis at the left or the scrollbar at the top.

You can zoom in/out by using the scroll wheel on the mouse.

By right clicking on the Y axis and selecting Zoom to fit, the plot area will be zoomed out so all available values fit. The timeline will be unaffected by this.

## Save measurement data

Measurement data can be saved to a data file and then opened and analyzed, by selecting the Save plotdata button and then selecting a file name.

## Tip

The readings are saved in a text file that you can import to Microsoft Excel® or another similar program for more detailed analysis.

## Open saved measurement data

To analyze measured data from a saved measurement, open a new diagram window and click on the Open plotdata button. Navigate in the file selector to the file you saved and click on Open. The readings are displayed in the diagram as they appeared when you saved them. Now use the various navigation options to look at and analyze the information. More on this is described in the section the section called "Analyzing using cursor lines".

## Save measurement setups

Setting up a measurement can be time consuming and the same set of signals are often reused. To make this easier measurement setups can be saved to be reloaded at a later time. To save a setup just press the Save Measurement Setup button and enter a suitable name.

## Open saved measurement setup

To reload one of the previously saved measurement setups just press Load Measurement Setup button and a drop down menu will be displayed. From this menu the desired setup can be selected.

### Analyzing using cursor lines

To analyze the measured graphs, there is the option of working with two vertical cursor lines that extend over the entire surface of the diagram. With the aid of the cursor lines it is possible to obtain a value for the time between two readings.

To show the cursor lines, press the Show cursors button in the toolbar. Move around the cursor lines with the aid of the mouse.

### Printout of screen view

To document the graphical image of a measurement you can print out the screen view. Press **Ctrl+P** or use the Print button in the main window toolbar.

# Downloading software

---

To get the control system going you have to download the compiled program files to the control units. The various control units must be loaded with various types of program file, depending on the unit's function in the control system.

This chapter describes how to proceed in order to download the various program files, and by following the instructions here you will have a system in operation in a short time. First there is a description of the main parts of the software. There is then a description of the two possibilities of downloading program files and how to proceed in these instances. But first of all, there is a short step by step reference for the most common use-case.

## Quick reference

### Downloading an application

The absolutely most common way to download software to the CANmaster control system is via the Terminal interface described in more detail in section the section called "Download of application program via Master/Terminal". Here is a step by step instruction how to use this to download an application that is ready for download.

To download an application to the CANmaster control system:

1. Open PC-Tool.
2. Open the project file either by double-clicking on the project file or from the File -> Open Project.
3. Make sure the RS-232 serial cable is connected to the Master or Terminal unit and the computer.
4. Press the Download button with the green arrow.



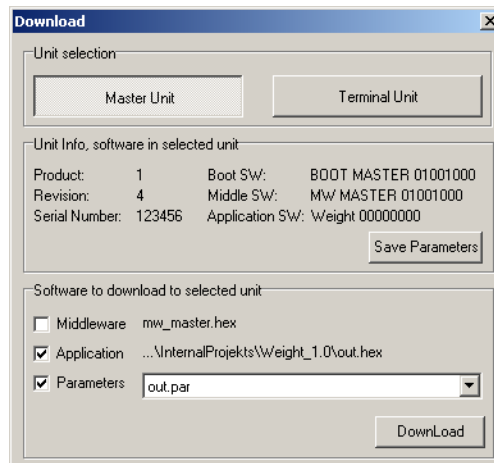


Figure 11.1. The dialogue box which is displayed when you wish to download software to the hardware via the Terminal or the Master unit.

5. Select the Master button at the top of the dialogue if it is not already depressed.
6. Make sure the Application and Parameters are checked. In case the units are newly delivered, also check the Middleware checkbox.
7. Press the Download button.

The download now starts and a progress bar is shown together with a window showing what is happening.

8. Close the dialogue via the Cancel button or the X in the top right corner when the download has completed.

## Uploading parameters

Another common task is to upload parameters from a system. For example, before any major changes done to the parameters of a system it is always wise to save a backup. Another use is if the master unit in a system is broken and the parameter settings are to be transferred to a replacement unit.

To upload the parameter settings from a CANmaster control system:

1. Open PC-Tool.
2. Open the project file either by double-clicking on the project file or from the File -> Open Project.
3. Make sure the RS-232 serial cable is connected to the Master or Terminal unit and the computer.
4. Press the Download button with the green arrow.
5. Select the Master button at the top of the dialogue if it is not already depressed.
6. Press the **Save Parameters** button

A dialog where a name and location for the uploaded parameters can be specified will be shown. Enter a suitable name and press **Save**.

The upload now starts and a progress bar is shown together with a window showing what is happening.

7. When the upload has completed, close the dialog via the Cancel button or the X in the top right corner. The saved parameters is now available for download by selecting them in the drop down box next to the parameter check box.

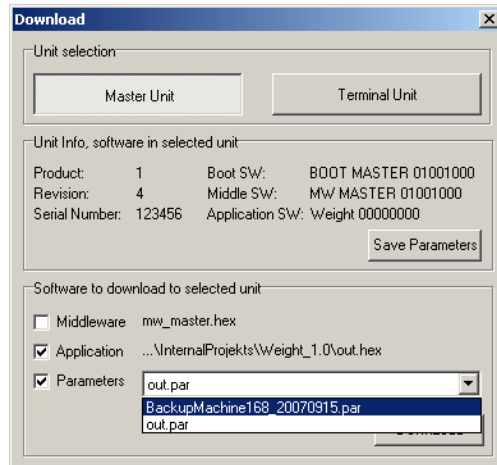


Figure 11.2. All parameter files present in the project folder can be selected for download. The file named **out.par** is the default parameter settings

## Subdivision of files

The control system's software comprises several parts that can be downloaded and updated separately. In certain cases the subdivision facilitates update of small parts of the system, thus obviating updating of everything every time an update takes place. Because the control system's various parts are interdependent you must make sure you use the correct versions of the program files you download to the units. Depending on which control unit a program file belongs to you can use various methods of downloading the program files.

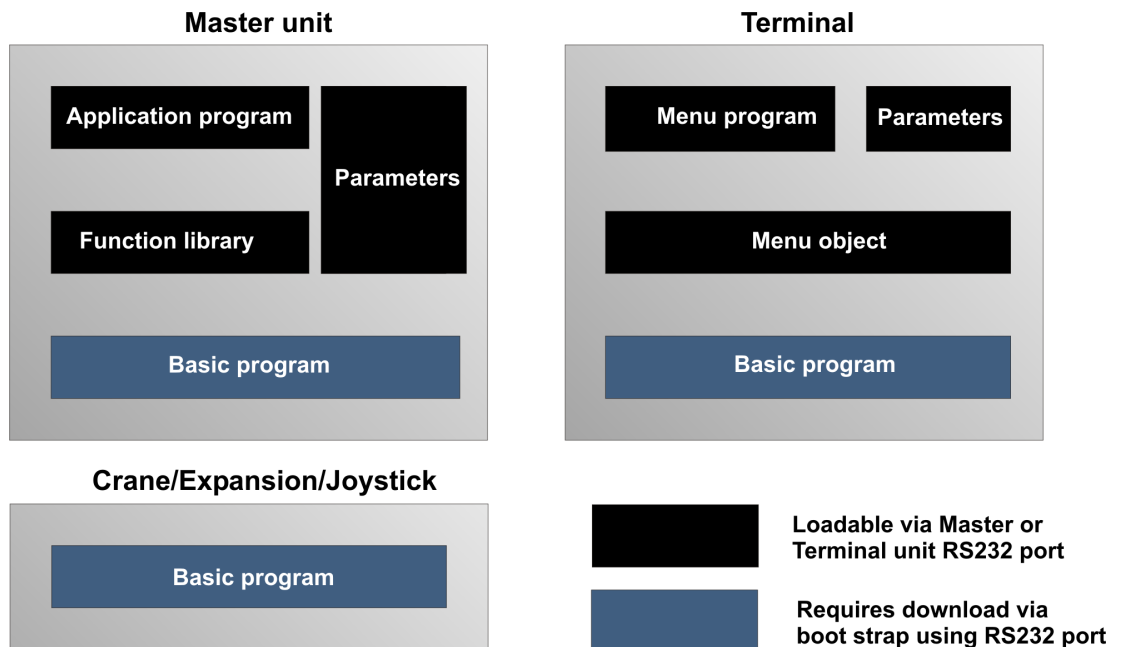


Figure 11.3. Schematic picture of the various program files required in the control units.

For the control system to function as intended it is necessary for all parts of the system to work with the same basic prerequisites. In this context it is important that the right versions of both software and hardware are used. Every part of the system - both software and hardware - has a version number and a

name that are used to verify that the various parts can work together. There is a possibility of downloading any program file at all in the system, but if the version numbers do not match the system will not start.

## Downloading various items of software

### Downloading application program and parameters

The program files that are usually affected by updates several times during a development project can be seen from the table below. All these files can be downloaded either via the Master unit's RS232 communication port or via the Terminal's RS232 communication port. It is of no significance which port is used. The CAN bus is used to automatically transfer the information to the correct unit if necessary. An RS232 D-sub contact or adapter that connects to RS232 must be permanently installed in the machine connected to either the Master unit's or the Terminal's RS232 communication port. Normally it is the Master unit which is connected to the machine's permanently installed communication contact.

Connection of the service computer to the system is via the computer's serial RS232 port using a straight serial cable (DB9M DB9F pin-to-pin) which is connected to the control system's RS232 port. If the computer has several serial ports you select the port to be used under ComPorts on the Settings menu. Your selection is saved so you do not need to select a port every time you are to use it to connect the computer to the control system.

The program files that are affected by updates in the normal work on the control system and that can be downloaded in accordance with the method above can be seen from the table below.

Table 11.1. Downloadable files.

File name	Comment
*.hex	Compiled application program for the system which is stored in the Master unit's program memory.
*.par	Compiled file with parameter information which is stored in the Master unit's parameter memory.
*.hydmmh	Compiled program for terminal menus that is stored in the Terminal's program memory.
*.hdr	Compiled file with information for the Terminal on where the variables affected by the terminal program are stored in the Master unit.

### Downloading basic software

All control units in CANmaster control systems are normally provided with all the requisite basic software on delivery, and handling of this is something you do not normally need to think about. Updating of basic software on supplied machines is very unusual, but it may be necessary if it is detected that a function critical to system safety is not functioning as intended. During a development project or in production it may be necessary to update certain parts of the basic software if there have been changes in the basic function library supplied so as to include these changes in ongoing production.

If updates to the basic software are to be implemented, new software must be downloaded to every single control unit affected by updating, and this is known as bootstrap. The service computer must then be directly connected to the relevant control unit's RS232 port, and in these instances there must be a prepared connection contact for the three wires that are connected to the RS232 port in the control unit. (See Technical Data & Installation User Manual).

Bootstraps with a direct connection to a unit can also be used to download only application programs and updated parameters to the Master unit and program files for new menu definitions to the Terminal. But normally downloads of these program files is via the fixed RS232 contact in the system regardless of the control unit intended, in accordance with the description in previous section.

### Download of application program via Master/Terminal

Select Tools -> Loader -> Terminal I/F... or click the Download button with the green arrow. When you do this, CANmaster PC-Tool will try to link to the connected unit. Then select the software you wish to download by first selecting control unit on one of the buttons in the upper part of the dialogue box, and then the files to be downloaded to the unit. If updating of the software in the Master unit is to take place you can select program library, application program or parameters. In the case of the terminal you can select download of the terminal's function library, menu definitions and relevant terminal parameters.

## Note

If you select download of a new version of the function library you must also update the software at application level, i.e. your application program or the terminal program for the application in question.

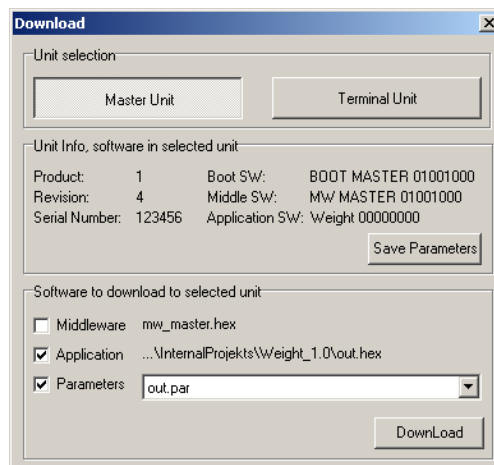


Figure 11.4. The dialogue box which is displayed when you wish to download software to the hardware via the Terminal or the Master unit.

When you open the dialogue box for download, CANmaster PC-Tool checks whether there is contact with either the Master unit or the Terminal. If CANmaster PC-Tool does not make contact with a unit the corresponding button in the upper part of the dialogue box will be deactivated and it will only be possible to select the unit which is available.

After a correct item of software has been selected for download, click on Download. Downloading of the selected software commences and a dialogue box showing what is happening is displayed. You can stop the download at any time by clicking on Abort, but after that the system will not start until you have downloaded a new application program to the hardware.

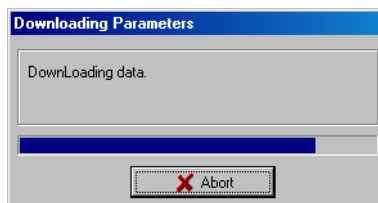



Figure 11.5. A dialogue box shows how the download is progressing. In this example only the parameter file is being downloaded.

## Download of program via bootstrap

You can use bootstrap to download all parts of the control system's software to every single one of the control units. It is also the only method that can be used to download updates to the basic software in all units.

Select Tools -> Loader -> Bootstrap... or click on the Download button with the red arrow . Depending on the unit to which you have connected the serial port, a dialogue box is displayed giving information on the software in question. If you have not yet connected a unit to the computer, a general dialogue box is displayed giving information on what to do.

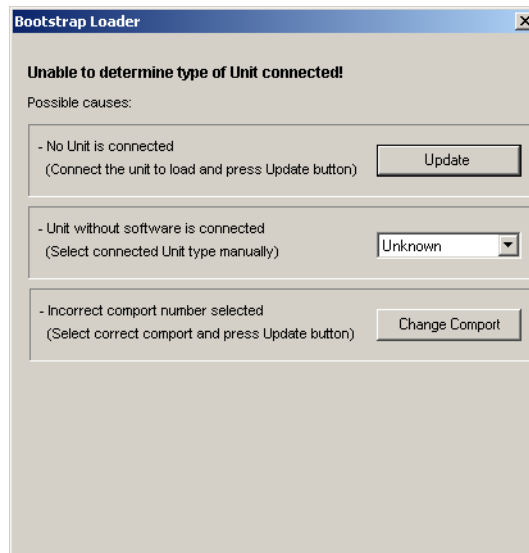


Figure 11.6. If you do not have contact between CANmaster PC-Tool and one of the control units, this dialogue box is displayed.

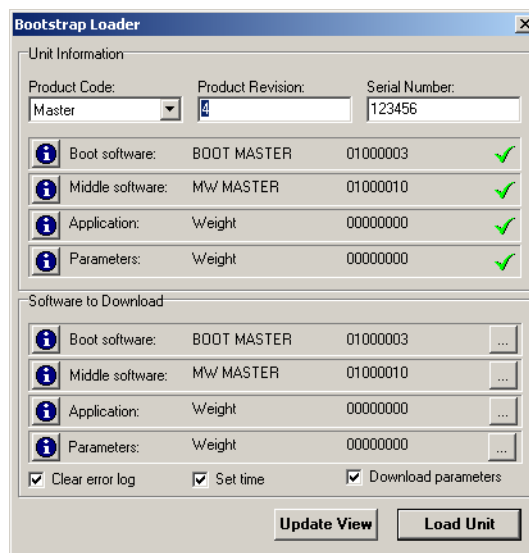
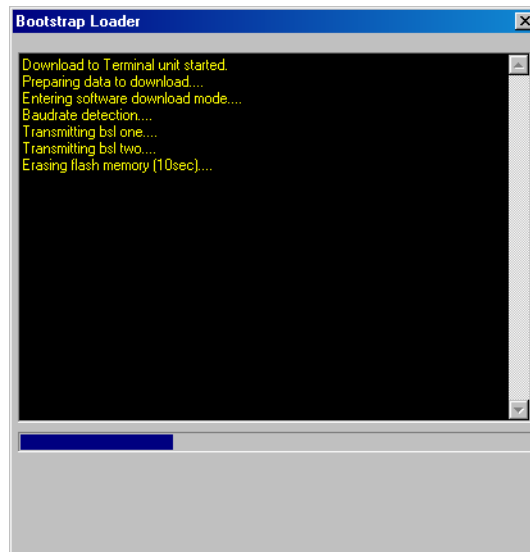


Figure 11.7. Download dialogue for the Master unit. In this example there is no software loaded in the unit. The lower part of the dialogue box shows which files will be downloaded to the hardware when you click on Load Unit button.

All the unit-specific dialogue boxes are divided into two different parts in the same way. At the top, information is displayed on the current software downloaded in the unit. Here you can see which version of the software is loaded, etc. By clicking on the Info button you get more information on every part of the software. Name, version, size and check sum are displayed there.



*Figure 11.8. Information on how the download process is progressing is displayed in a dialogue box whilst the download is in progress.*

After you have selected download of a specific file to the hardware, a dialogue box is displayed giving detailed information on what is happening. An indicator also shows what proportion of the entire process has been completed. The information displayed in the window is also written to the log window under the Download tab. In the event of a fault it is thus possible to see what is happening and how far the process got before a problem arose.



## Error reports

---

In a control system errors of various kinds can always arise. There can be internal electronics errors in the control units, errors in the basic software, design errors in application software, exceeded limit values for sensor signals or direct errors in external components such as short circuits and poor contact or impermissible levels of supply voltage for the various parts of the control system.

To facilitate error management in the control system, all types of error message reported in any part of the system are saved in a central error log placed in the Master unit. All errors occurring in the system are automatically transferred via the CAN bus to the Master unit's central error log. Every unit also has its own internal error log where the unit specific errors are saved. All error logs in the system are saved in the EEPROM memories, where the information remains when the system power supply is turned off.

The information in the Master unit's error log can be uploaded to a service computer via the unit's RS232 communication port for display on-screen and documented by print-out or saved in a separate file. The error log can also be uploaded to the terminal's display, but in this case only overall information on the errors is stipulated, in the form of a code number.

By each unit having its own error log the local errors can be analyzed even if the unit is not connected to the system, thus facilitating servicing and troubleshooting of separate control units. It also guarantees that a local error does not disappear if the automatic transfer to the Master unit via the CAN bus does not work because of a cabling error, electronics error or error in the software that handles the communication on the CAN bus.

The registered errors that are stored in the Master unit's error log contain comprehensive information on every error that has occurred, to facilitate easy tracing of the cause of the error, with information on the type of error, the unit in the control system that generated the error and the date and time when the error occurred (time stamp).

### Uploading error logs

To upload the error log from a control unit the service computer is connected to the unit's serial RS232 communication port. Setting of the communication port on the computer to be used is indicated via Settings -> ComPorts. When the correct setting has been made you should select Tools -> Errorlog to upload the error log to CANmaster PC-Tool.

When you open the error-log window in CANmaster PC-Tool the program automatically tries to connect to the unit linked to the computer's communication port. If the program cannot find a connected control unit, an error message is displayed stating 'connection to a control unit missing'. When contact has been set up, the types of uploadable error log are displayed. If the service computer is connected to the Master unit or to the terminal, then the error logs in both of these units are available to be uploaded, regardless of the unit the service computer is connected to. By selecting the required type from those available in the box for control units and then clicking on Upload, the selected error log is uploaded to CANmaster PC-Tool and displayed in the error-log window in the form of a table. Every time you click on Upload new information is placed at the end of the list of error logs.

To upload an error log from a control unit to CANmaster PC-Tool do as follows:



1. Select the required type of control unit from which the error log is to be uploaded.
2. Click on Upload to upload the error log to CANmaster PC-Tool.

To clear a saved error log in a control unit's memory, do as follows:

1. Select the available error log by highlighting the radio button belonging to the required unit.
2. Click on Clear to clear the error log in the control unit.

To delete the content of the table in the error-log window without deleting the error log in the control unit, close and re-open the error log window in CANmaster PC-Tool.

## Error-log window

The error-log window displayed in CANmaster PC-Tool presents uploaded error logs in the form of a table in which the error in question takes up one line. The error that occurred last is displayed on the top line. Every error message has a serial number which is indicated in the column with the character '#'. The other columns indicate the following information for the error message in question:

Date	Date when the error occurred.
Time	Time when the error occurred.
Unit	The unit that generated or detected the error.
Type	Type of error.
User Action	Event; there is an error or the error has disappeared.
SW	The software's execution mode when the error arose.
Error Data	Other information on what caused the error, stipulated in plain text.

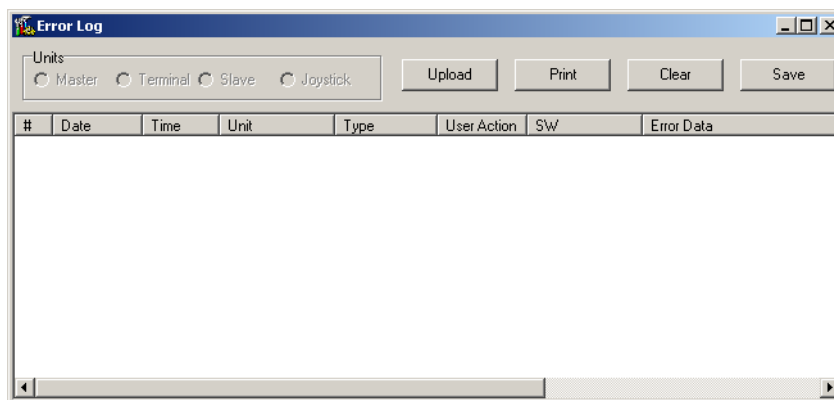


Figure 12.1. The error-log window in CANmaster PC-Tool presents an uploaded error log in tabular form in which every error takes up one line in the table, with the latest error on the top line.

### Description of the information in the error-log window

There follows a more detailed description of what is indicated in the relevant column in the error-log window's table.

#### Column - Unit

The unit that has detected an error is displayed in the Unit column. MASTER, TERMINAL, CRANE #n, EXPANSION #n or JOYSTICK #n are reported here. If the system includes several units of the same type, the unit's number is reported after the unit type.

## Column - Type

The information in this column indicates the type of error that has occurred.

The following are the three possible types of error:

**ALARM** Errors stipulated by the designer of the machine's application program that should be reported as 'errors'. For example, this applies to incorrect sensor values or cabling errors.

**EMERGENCY** Errors in the system's internal functions. These errors are of the same type as is defined in the CANopen standard, and can for example be voltage errors in the units, short circuits or broken communication.

**ERROR** Errors in the internal operating system in the control unit in question. These errors do not arise if the system is working normally. But if incorrect application software has been downloaded or another similar problem has arisen, this is reported as an ERROR. Another error of this type is full occupation of the memory areas - something which cannot occur if the application program has been correctly designed.

## Column - User Action

This column registers whether an error exists or whether it has ceased, i.e. it gives some history of the error. Every single one of these events is reported as a separate error message. When the error occurs it is registered with `RAISE` and if it disappears with `CEASE`.

## Column - SW

The information states the mode the system is in when the error is detected. Normally `APPLICATION` is given, which means that the control system is working normally with the application program in operation. Less common is `BOOT`. In that mode only internal system errors are reported, since the application program is not in operation.

## Special errors that occur upon system startup

Information in the error log on date and time is based on the time stipulation applying locally in the individual control unit in which the error is registered. To synchronise all units in the system, time counting is controlled by a common system clock placed in the Master unit. The Master unit issues a time signal once a minute on the CAN bus. If an error message in a unit has the time value set to zero, this is because an error has been registered in the unit before the first time signal has come following system startup. Errors of this type should not normally occur, and they reveal an error in the application program or in the connection of the control system.

# Saving error logs

An uploaded error log can be saved to a data file for documentation and further follow-up. The file is saved in an ordinary unformatted text file and can thus be opened in a computer that does not have CANmaster PC-Tool installed.

The error log is saved to a separate file by selecting File -> Save or clicking on the Save-button in the toolbar.

The information is saved in columns of a fixed width and the file can be opened in an ordinary text-editing program such as Notepad or Wordpad, or imported to a calculation or database program for further processing. The files names end in `.log`.

You can also select to print out the table displayed in the error-log window. This is done by selecting File -> Print.... Then select printer and stipulate the print-out format 'landscape' so as to get unbroken lines. The column headings are printed out at the top of each page.

# Internal error messages

This section lists all the possible basic internal error messages (except the application-dependent error messages), giving a detailed description of the error message in question, possible cause of the error

and proposed measures for rectifying the error. These errors are not presented on the terminal's display but are only stored in the control units' error logs. To check which errors of the type below have been reported, the error log must be uploaded to the service computer. The central error log can be uploaded to the terminal if the error is of such a type that the terminal functions are still working, but just show an overview of information on what has been reported on in the error log in the form of a code number.

All types of error also have a stipulated degree of risk level for continued operation. The errors stipulated as having the level 'Critical' are system errors, and are dealt with by the system in a special way. These errors must not occur more often than three times every 30 minutes. In the event of a higher frequency the control system is automatically switched off. Before the system is taken back into operation the error log should be carefully analyzed and any causes of the error rectified.

The various errors that can occur in the system belong to one of the following groups:

- Emergency
- Voltage & Temp
- Joystick
- CAN-bus
- Error

## Emergency

**EMERGENCY** errors are often related to the actual installation of the control system or to the electronic circuits in the control units. The following error messages are in this group:

No Error or Reset

Indicates that the reported event does not constitute a direct error. This event can be reported after a restart of the system as a result of a previous error. Ordinary restarts of the system are not usually reported, but if a system error has occurred that has caused an automatic restart, then this event is reported.

Generic Error

Current Error

Current Error device input

Current Error inside device

Current Error device output

Voltage Error

Voltage Error Mains

Voltage Error inside device

Output Voltage Error

Temperature Error

Ambient Temperature Error

Device Temperature Error

Device Hardware Error

Device Software Error

Device Software Error Internal  
 Device Software Error User  
 Device Software Error Data  
 Error additional Modules  
 MONITORING ERROR  
 Communication Error  
 Communication Error CAN overrun  
 Communication Error CAN passive mode  
 Communication Error LIFEGUARD or HEARTBEAT  
 MONITORING RECOVER BUSOFF  
 MONITORING ERROR TRANSMIT COB ID  
 PROTOCOL ERROR  
 PROTOCOL ERROR LENGTH ERROR  
 PROTOCOL ERROR LENGTH EXCEEDED  
 EXTERNAL ERROR  
 ADDITIONAL FUNCTIONS ERROR  
 DEVICE SPECIFIC ERROR

## Voltage & Temp

Unit-specific error messages as a result of internal or external voltage levels are reported as follows:

Voltage Error - VDD

**Level:** Critical

Insufficient supply voltage for the drive stages that drive the external outputs. The error is usually a result of short circuits or poor cabling between the control unit and the valves connected to the outputs. The error must be rectified before the system can be put into operation.

Voltage Error - V-Sys

**Level:** Critical

The main voltage for the system is under 11 volts - the lowest permissible voltage on which the control system can function. The error may be the result of external circumstances, e.g. short circuits in sensors or valves connected to the system. The error can also be the result of internal problems in the control unit. Another cause can be poor capacity of the machine's batteries, or the generator not loading sufficiently.

The error must be rectified before the system is put into operation.

Voltage Error - V-Out

**Level:** Critical

Insufficient internal voltage for digital outputs and PWM outputs. The error must be rectified before the system is put into operation.

Voltage Error - VREF-OUT

**Level:** Critical

Insufficient voltage on output Vref, which supplies sensors and potentiometers. The voltage must be 5.0 volts, and a warning is issued if the voltage level is under 4.8 volts. The commonest cause of this error is faulty cabling or incorrectly connected sensors. The error must be rectified before the system is put into operation.

Voltage Error - VSS

**Level:** Critical

Incorrect level of internal system voltage. Contact the supplier of the control system for error reports and any replacement of the control unit if the error is recurrent.

Voltage Error - V-CAN

**Level:** Critical

Insufficient supply voltage in the CAN bus. The problem usually arises in the event of poorly insulated cabling or loose connections in the CAN cabling. The error must be rectified before the system is put into operation.

Voltage Error - VCC

**Level:** Critical

Incorrect level of an internal system voltage. Contact the supplier of the control system for error reports and any replacement of control unit if the error is recurrent.

Temperature Error - Low

**Level:** Critical

The operating temperature has gone below the limit value for the unit's lowest permissible operating temperature. Rectify the operating environment so that the operating temperature is within the specified area. The error must be rectified before the system is put into operation.

Temperature Error - High

**Level:** Critical

The limit value for the unit's maximum permissible operating temperature has been exceeded. Rectify the operating environment so that the operating temperature is within the specified area. The error must be rectified before the system is put into operation.

Short cut Error

Digital Output #n

**Level:** Critical

There is a short circuit on an output. If the error is detected on a digital output the report will state the output(s) in question. The error must be rectified before the system is put into operation.

## CAN-bus errors

In the event of a communication error in the CAN bus the Master unit reports the following error:

Communication Error - MASTER

A problem has arisen in a unit in communication with the Master unit. This can be the result of a loose contact or other problems in the cabling for the CAN bus. The error can also be reported if the supply voltage to the Master unit is interrupted at the same time as other control units in the system have the correct supply voltage. Connected control units switch off automatically when this type of error arises. Check that the cabling and that all control units in the system have the correct supply voltage.

Communication Error - TERMINAL

The Master unit has lost contact with the terminal. Errors of this type are usually the result of errors in cabling or the connector.

**Communication Error - JOYSTICK #*n***

The Master unit has lost contact with the stipulated joystick control. Errors of this type are usually the result of errors in cabling or in the connector.

**Communication Error - EXPANSION #*n***

The Master unit has lost contact with the stipulated Expansion unit. Errors of this type are usually the result of faulty cabling or a faulty connector.

**Communication Error - CRANE #*n***

The Master unit has lost contact with the stipulated Crane unit. Errors of this type are usually the result of faulty cabling or a faulty connector.

## Joystick unit

Error messages from a Joystick control can contain the following information:

Illegal interrupt vector. Address=*0xNN*

**Level:** Critical

Invalid interruption vector. The interruption vector used to handle various processes and events in the joystick control's software is incorrect.

Checksum error

**Level:** Critical

Incorrect checksum. The software in the joystick control is checked upon starting up the system and regularly during operation against a checksum calculated in advance. If these values do not tally, the error 'checksum error' is stipulated. Errors of this type are usually the result of errors in the software, incorrectly downloaded software or problems with the program memory in the joystick control.

Bit Error RAM

**Level:** Critical

Faulty memory. The program in the joystick control has checked the system memory and has detected that parts of the memory are not working.

Bit Error ERAM

**Level:** Critical

Faulty memory. The program in the joystick control has checked the system memory and has detected that parts of the memory are not working.

Invalid parameter value. ParameterIndex=*0xNN*

**Level:** Critical

Incorrect value upon updating parameters. An update of parameters failed, since the new value was incorrect. The parameter value is outside the permissible limits, or wrong parameters were updated. Check the parameter values and try again.

Error in errorlog

**Level:** Critical

Incorrect index in the error log. The joystick control's software has detected an incorrect value in the internal error log.

Critical Error in SW. Address= *0xNN*

**Level:** Critical

A critical error has been detected in the joystick control's software. The problem is at the stipulated address. The joystick control must be sent in to the supplier for troubleshooting and repair.

#### WatchDog

**Level:** Critical

The joystick control has got stuck in a non-permissible mode in the program and has automatically restarted. If the error is recurrent there may be an error in the software, or incorrect software may have been downloaded to the control. Check that the correct version of the software is being used in the joystick control. If no software error can be established the Joystick unit must be sent in to the supplier for troubleshooting and repair.

#### Parameter changed ParameterIndex=0xNN

**Level:** Information

A configuration parameter has been changed. This does not always constitute an error but can be used for checking of various changes previously made in the system when other errors have occurred.

#### Error Joystick X-axis

The joystick control's signal values on the X axis have been reported as being outside permissible limit values. The reason may be an error in the potentiometer for the X axis or other mechanical errors in the Joystick unit. If the error recurs the Joystick must be sent in to the supplier for troubleshooting and repair.

#### Error Joystick Y-axis

The joystick control's signal values on the Y axis have been reported as being outside permissible limit values. The reason may be an error in the potentiometer for the Y axis or other mechanical errors in the Joystick unit. If the error recurs the Joystick must be sent in to the supplier for troubleshooting and repair.

#### Error Joystick Z-axis

The joystick control's signal values on the Z axis have been reported as being outside permissible limit values. The reason may be an error in the potentiometer for the Z axis or other mechanical errors in the Joystick unit. If the error recurs the Joystick must be sent in to the supplier for troubleshooting and repair.

#### Reference voltage Error

Incorrect supply voltage for the Joystick unit has been detected by the software in the control. Check all connections and the level of the supply voltage, and make sure the Joystick is correctly connected. In the event of recurrent problems the entire installation for the Joystick must be checked and perhaps also the Joystick unit's internal electronics.

#### Digital voltage Error

Incorrect voltage for the digital outputs has been detected. Check all connections to make sure there are no short circuits in the cabling or other errors in the connector. If the problem persists the entire installation must be checked. The Joystick unit may need to be sent in to the supplier for troubleshooting and repair.

#### Supply voltage Error

The supply voltage for the joystick control is outside stipulated limit values. Check the system's supply voltage and check that all connections are correctly implemented.

## Error

Type=N, Count=N, Extra=0xNNNNNNNN, IP=0xNNNNNNNN, CSP=0xNNNNNNNN, PSW=0xNNNNNNNN, PCB=0xNNNNNNNN

This type of error message is the result of internal errors in the basic software. Make a note of all the information stipulated in the error message and, as far as possible, the course of events when the error arose, the functions that were activated and the machine's operational mode. Forward the information to your system supplier for closer analysis and troubleshooting.

The figure stipulated after `Type` gives more precise information on the cause of the error, and is very important to the continued troubleshooting work.

---

The figure after `Type` has the following meaning:

1. Invalid vector (invalid interruption). An error of this type states that an error has arisen in the downloaded software or that the wrong version of the software has been downloaded. The value after `Extra` is important information for the system developer, providing information on which vector is incorrect.
2. The memory stack in the system has been exceeded or the stack has been fallen short of. Value = 0 indicates that there is a stack overflow, whilst Value = 1 indicates that the stack has been fallen short of. Problems of this type are often the result of wrongly designed application software. Check that the application program has been correctly designed.
3. Invalid instruction. This type of error means that an impermissible instruction has been executed. Either the basic software or another part of the software is incorrect, or incorrect software has been downloaded to the system.
4. Error in the electronics. A general error in the electronics has been registered. The unit must be sent in for troubleshooting and repair.
5. Memory error. An error in the memory in the control unit has been detected. The information after `Extra` indicates the address in the memory containing the error. The control unit must be sent in for troubleshooting and repair.
6. Invalid system call. This error indicates that the application program or function library has performed a prohibited measure via a system call to the operating system in the control unit. Incorrect software in the function library or the application program has been downloaded and must be updated.
7. End of message. The error indicates that a message that has been sent has ended before receipt of the whole message, thus pointing to communication problems. Check all connections. If the error recurs the control unit must be sent to the supplier for troubleshooting and repair.
8. Faulty message. A message that has been sent to the control unit has had an incorrect format and can thus not be deciphered. Incorrect basic software has probably been downloaded to a control unit. Download correct software and test the system again.
9. End of queue. A message queue has received an enquiry about looking after a further message, but there is no room for it in the current queue. The problem is probably the result of communication problems or over utilisation of the memory resources in the control unit. Check the application program and check that communication between all connected units is working.
10. Incorrect queue. A message has been sent to a non-existent queue. The problem is the result of wrongly downloaded basic software or a wrong application program. Make sure the right versions of all software are downloaded and try again. If the problem persists the control units must be sent in to the supplier for troubleshooting and repair.
11. Incorrect process. An internal process in the control units is incorrect and will thus not be executed. This means that some functions doesn't working as intended. The error is probably the result of compatibility problems or errors in downloaded basic software. Download the correct versions and test the system again. If the problem persists the control unit must be sent in to the supplier for troubleshooting and repair.
12. Invalid event. A registered event or message is incorrectly formatted and cannot be interpreted by the basic software. The error is probably the result of incorrect software. Check that the right software is available and reload the software to all control units. If the problem persists the control units must be sent in to the supplier for troubleshooting and repair.
13. Incorrect configuration. The basic software has detected a system configuration it does not support. The incorrect basic software or application program has probably been downloaded to the system. Check that the right software is downloaded in the system and restart the system. If the error persists the control units must be sent in to the supplier for troubleshooting and repair.





# Index

---

## B

Bootstrap, 87

## D

Download

- application program, 86
- basic software, 86
- com port, 86
- parameters, 86
- via Master/Terminal, 86

## E

Execution order

- general, 27
- global, 28
- local, 28

## F

File types

- .hydcf, 17
- .hydmd, 18
- .hydsr, 17
- .mtm, 16
- .src, 17
- out.hdr, 19
- out.hex, 19
- out.par, 19
- out.sym, 19

## M

Monitor

- scale factor, 81

## P

Parameters

- function parameters, 53
- out.par, 59
- system parameters, 53
- unit specific parameters, 53

Project.hydrj, 15

## R

Revision control, 22

## T

Terminal I/F, 87

