Eurotherm®

by Schneider Electric

LINTools
Engineering Studio

User Guide

HA263001U055

September 2017 (Issue 19)

LINTools Engineering Studio

User guide

Table of Contents

1. LINTOOLS ENGINEERING STUDIO

LINtools is a powerful multi-purpose Windows-based software package for use both off- and on-line.

- **Offline**, LINtools creates and modifies a Local Instrument Network (LIN) based process strategy, sequences, and actions for a range of target instruments. It is also used to configure instrument properties, e.g. time-zone, time sync, protocol name, Alarm Suppression, etc.
- **Online**, allows the user to monitor and interact with control and sequence strategies running in remote instruments across ELIN or ALIN, providing a versatile commissioning and strategy debugging toolkit.
- **Online Reconfiguration**, allows the reconfiguration of an operational instrument via an Ethernet network.
- **I/O Configuration**, LINtools creates and modifies a block-structured I/O subsystem.
- **Communications Configuration**, allows the user to configure the Data Exchange requirements third party devices, e.g. Modbus, Profibus, etc.
- **Data Recording Configuration**, allows the user to record values from defined groups of the parameters, and configure the appearance of each defined parameter when shown in Review. Additional configuration of the Instrument Properties provides an archiving strategy for the recorded values.
- **Setpoint Programming Configuration**, allows the blocks required as an interface to the Programmer Editor to be created. The blocks are created with a default configuration, but are used to display the configuration of the Programmer Editor when online.
- **Intellectual Property Protection**, allows specific application file types to be encrypted to prevent the loss of Intellectual Property.

LINtools uses and outputs fully compatible files that can be saved locally to hard disk and transferred to and from target instruments at high speed over the LIN via ELIN (Ethernet), or for legacy instruments.

Designing a block-structured strategy and sequence with LINtools is fast and straightforward. Engineering productivity is boosted by its ability to function as a central configuration terminal for all our LIN-based products, capable of building a global database for distribution across the product range - both current and future.

1.1 RELATED MANUALS

1.1.1 LIN Blocks Reference Manual (HA082375U003)

This manual describes the available instruments function blocks that can be connected to this manufacturers Local Instrument Network (LIN). The purpose and workings of each block are explained, specification menu parameters are defined and inputs and outputs indicated, giving enough detailed information needed to configure the function block in a Strategy.

1.1.2 Application And Control Modules Operator Manual (HA084012)

This manual describes the application and control modules available within the LIN environment. These modules can be used in conjunction with existing LIN blocks to perform a wide range of control tasks. Application modules can be utilised for specific activities such as combustion control, load management and pump set control, while Control modules consist of common devices such as motors and valves.

Note: Please contact your supplier if these documents are not available.

2 OVERVIEW OF LINTOOLS

LINtools is a stand-alone application that can also operate as part of a Project based suite of tools. It is used to configure the strategy (operation) of an Instrument that communicates with other nodes via a Local Instrument Network (LIN), i.e. other LIN Instruments and Computers. It provides graphical configuration for continuous control and sequencing control using function block configuration, and sequential function charts (SFCs) and ladder logic, allowing the creation of actions and steps.

LINtools offers:

- A common toolset for the LIN product range.
- Intellectual Property Protection.
- Graphical configuration.
- An extensive library of function blocks and control modules.
- Powerful online configuration features for instrument strategy files that may or may not exist in Instrument Folders in a Computer based Project.
- Powerful online commissioning features.
- Automatic documentation.

2.1 THE LINTOOLS COMMON TOOLSET

LINtools is a comprehensive set of configuration, test, documentation, and commissioning tools for LIN instruments. LINtools employs graphical configuration for continuous control and sequencing control.

The online reconfiguration features give configuration engineers the ability to a change Strategy while running; not only to change parameters, but to change the structure of the Strategy, to allocate new elements in memory.

Note: The Tools menu on the Menu bar provides access to applications that can be launched from the LINtools Engineering Studio. These applications are used to enhance the operation of the product range.

The online connect facility allows configuration engineers to view running Databases and LIN Sequential Function Charts (SFCs) - making for easier commissioning and trouble-shooting of a Strategy.

Note: The Connect feature in LINtools provides the functionality to monitor values from the database of a ‘Live’ Instrument.

2.2 GRAPHICAL CONFIGURATION

- Both Continuous and I/O strategies are configured graphically as a Function Block Diagram (FBD). FBDs use ‘block-structured’ technologies provided by the LIN Database Editor, supporting a comprehensive library of LIN function blocks.
- A LIN Sequence is configured graphically using Sequential Function Charts (SFCs) following the IEC1131-3 standards. Steps initiate Sequence Actions - which may be Structured Text (ST) statements, Ladder Diagrams, or nested SFCs. Transitions determine when control passes from one Step to another.

Sequences (SFCs) can be configured as Generic or Specific. Generic Sequences let the user configure and test the Sequence once, and then replicate it many times. Generic Sequences are constructed using generic name variables. The Sequence Editor allows the creation of one kind of Sequence from the other, i.e. Specific Sequences from Generic Sequences, and vice-versa.

2.3 FUNCTION BLOCK AND CONTROL MODULE LIBRARY

In addition to its extensive library of function blocks, LINtools offers a comprehensive library of standard control modules to provide a ready-made set of versatile building blocks. The standard control module library includes a wide range of valves, motors, and signal conditioners. Control

modules enable plant devices and equipment to be represented by dedicated objects in the control system with standard displays at workstation level, that

- simplifies implementation
- clarifies documentation
- helps operators navigate around the displays

2.4 ONLINE RECONFIGURATION FEATURES

The Online Reconfiguration features allow the user to access the LIN Database files, .dbf, within a LIN Instrument.

By using the Connect feature, a user can access a LIN Database file, .dbf, in a 'Live' Instrument, without having the files contained in the Computer based system. However, once connected to the 'Live' Instrument, the Strategy files can be inspected, edited (via Online Reconfiguration), and even uploaded to the Computer based system.

By using the Reconfiguration features, including Try, Untry, and Apply, a user can tentatively try changes to a LIN Database file, .dbf, in a 'Live' Instrument. In a Non-Redundant (Simplex) system or a Redundant (Duplex) unsynchronised system the changes can be tried to assess the impact these may have on the Strategy. When the user is satisfied with the changes that have been made, they can be saved (applied) to the operational system, and Project environment using a single ('Apply') button.

Note: Any instrument operating in Duplex mode, with a synchronised secondary processor, MUST be desynchronised to permit Online Reconfiguration. The secondary processor must be resynchronised when Online Reconfiguration is complete.

2.5 ONLINE COMMISSIONING FEATURES

LINtools Connect and Monitor facilities allow operators to view and interact with Databases and Sequences running in remote instruments on the LIN.

- With the LIN Sequence Action Monitor and the necessary security clearance, remote Sequences can be observed and controlled, active steps can be tracked, Transitions can be forced or held, and SFC objects can be inspected.

2.6 INSTRUMENT FILE SYSTEM

Instrument file names are displayed in a simplified Contents Pane, which initially shows all files in the Instrument Folder that will be downloaded to the Instrument.

Any file not used in the Strategy is stored in the 'Unused Files' folder, but can be added to the strategy at any time. This folder can be shown or hidden using the Show Unused Files item in the Contents pane context (right click) menu.

Note: Some files types are grouped for simplicity, but can be individually added to the list of files to be downloaded to the Instrument.

2.7 AUTOMATIC DOCUMENTATION

The Strategy and graphics can be fully documented using the Documentation utilities and transferred across the network using the File utilities.

LINtools follows the **IEC1131-3** programming standard. It is the worldwide standard for the programming interface in industrial control systems for a wide range of applications, supporting Function Block Diagram [FBD], Sequential Function Charts [SFC], Ladder Diagrams [LD] and Structured Text [ST] programming languages.

3 CONFIGURING A STRATEGY - OVERVIEW

The following stages may occasionally overlap, and their order can be varied, especially when an existing strategy is being edited. Function blocks, wiring, parameter values and Sequences can be added, modified or deleted at any time using the LINtools toolkit, even if the Strategy files do not exist in a Computer based Project.

Note: If a Strategy already exists on a 'Live' instrument in the plant/system, use the 'Connect' feature to select the Instrument and either edit the Strategy online, using online reconfiguration, or upload to the Computer, edit offline, and then download the changes to the target instrument.

To configure a Strategy, follow these stages:

1. For a new Strategy, first create and configure a LIN Database.

   By using the New Instrument wizard (or automatic database generation if supported by the instrument), all essential blocks are automatically added to the database file. These blocks include an. Instrument Configuration Block (Header block) describing the Instrument Type, Version, and fields to determine the instrument configuration, and a pre-defined range of Diagnostic Blocks, used to assist with diagnosing potential or existing faults.

   A LIN Database processes inputs to control the behaviour of an actuator, or plant instrument, in a required manner. If a LIN Database already exists in the 'Live' LIN Instrument, use the Connect feature to access the existing Strategy files.

Note: A LIN Database always requires a Header block (Config category) to determine the Strategy instrument type.

2. Then, create and configure a LIN Sequence.

   A LIN Sequence defines a list of steps when the process being controlled by the LIN Database can adopt several distinct states.

3. Monitoring across nodes may be required, so create and configure a LIN Monitor Database.

   A LIN Monitor Database is only required if you wish to review a selection of blocks from within a number of different nodes. For a single node use the Connect facility.

Note: Any Sequences operating in the selected LIN Monitor Database will also be monitored, if the SFC_CON and SFC_DISP blocks have been added to the LIN Database.

4. Next, create and edit a LIN Action.

   A LIN Action defines direct operations on the LIN Database of the running Strategy.

Note: Some application files can be encrypted in order to protect your Intellectual Property. Set 'Offer option to encrypt files on Save As' checkbox  on the Advanced page of the Settings dialog to allow you to configure password protection.

The strategy is now complete and ready to be downloaded to the instrument.

4 CONTENTS PANE

The Contents pane permits the creation, editing and inspection of LIN Databases, LIN Sequences and Actions. The links on this page reveal instructions concerning how to use its features.

The default view of the Contents Pane shows a list of the files used in the Strategy, via a tree view structure. Any file displayed in the default view of this pane can be opened in the appropriate application and will be downloaded to the Instrument.

Notes:
1. Each of the panes, Template palette, Object Properties, etc. can be individually docked anywhere in the application window, e.g. the Template Palette can be docked below the Contents Pane.
2. Use the <Ctrl> key while moving the pane to prevent it from docking unintentionally. This allows it to move freely around the Computer window.

The icons displayed in the Contents Pane allow specific operations to be applied.

Note: Some files are have been grouped for simplicity, e.g. .dbf, .grf, .dtf, but can be individually added to the list of files that are to be downloaded to the Instrument.

4.1 WHAT IS THE CONTENTS PANE?

The Contents Pane is the list of files contained in the selected instrument. Its purpose is to control and manage all the files that together form the Instrument Strategy and any other files that are used to enhance the operation of the Instrument, e.g. displays created using the User Screen Editor, and gateway files created using the Modbus tools. The pane is designed to offer the various file types and their own set of context sensitive menu commands.

The Toolbar at the top of the Contents pane offers a quick access to the most common commands that are used regularly in an Instrument context.

4.1.1 Use of the Contents pane

The Contents pane lists the files that are currently part of the strategy and can be selected for downloading to the instrument.

The files stored in the Instrument Folder can be controlled by using the Contents pane toolbar. These toolbuttons are used to allow files to move freely between the existing folders. An Unused files folder remains hidden unless specifically requested via the context (right click) menu.

If the Contents pane is not visible, select 'Contents' in the View menu. The ▾/▴ symbol can be used to minimise or maximise the Pane if more than one pane is docked in the same area of the application, and the ✕ can be used to hide this pane.

Note: Use the <Ctrl> key while moving the pane to prevent it from docking unintentionally. This allows it to move freely around the Computer window.

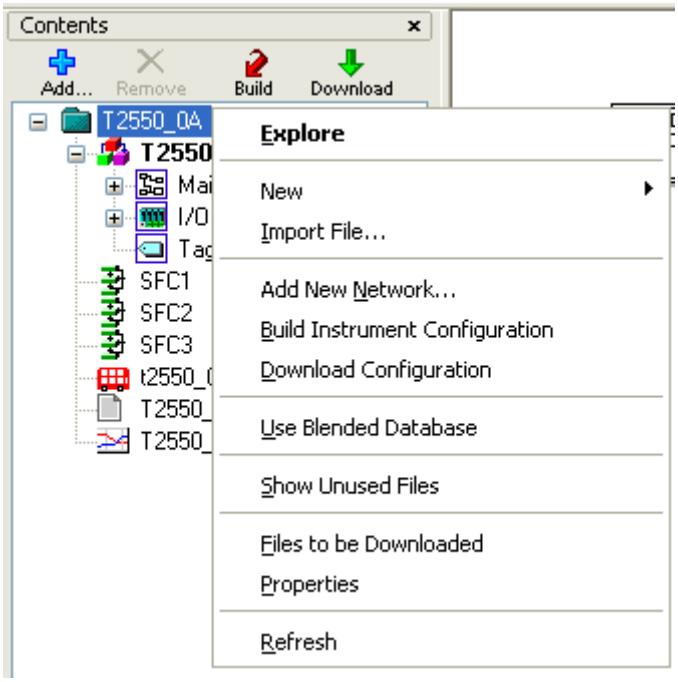
- Open sub-windows can be cycled through using the editor's Back and Forward toolbuttons.

Caution

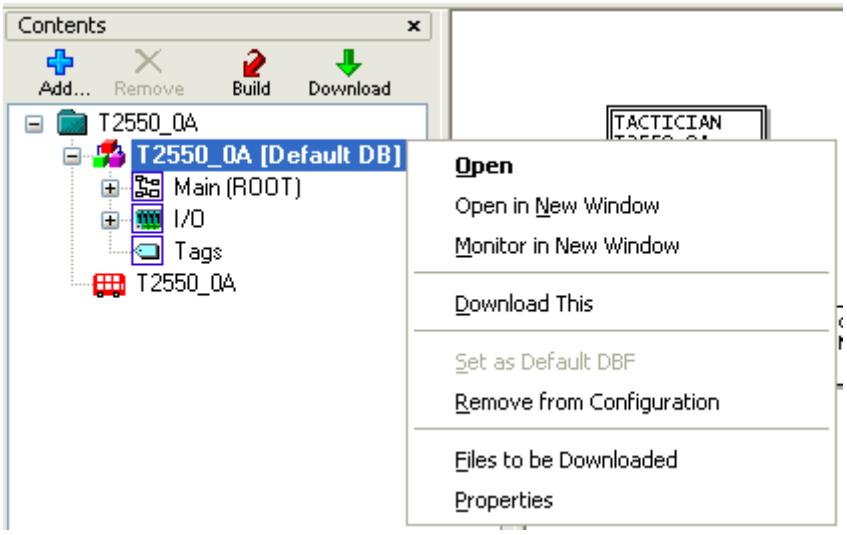
Any changes made while online may seriously affect the operation of the system. While performing offline changes to instrument blocks in the LINtools Strategy, the states of the Node, Module, and/or Channel blocks are updated when saved.

Note: During Online Reconfiguration any changes made to the I/O Configuration are indicated by the icons displayed in the Content pane.

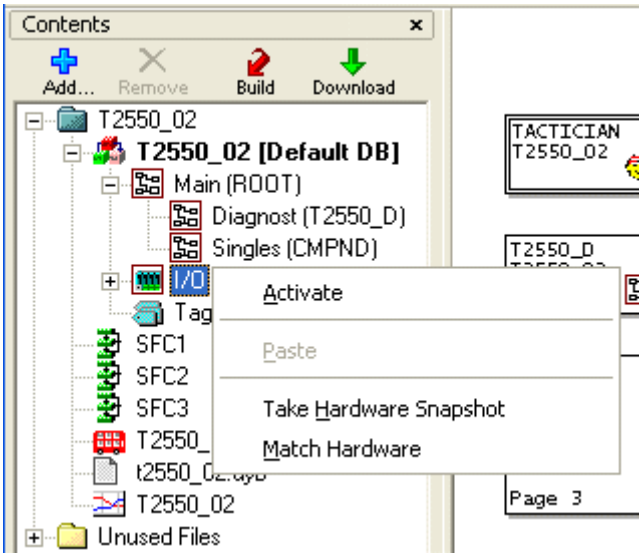
4.2 THE INSTRUMENT FOLDER



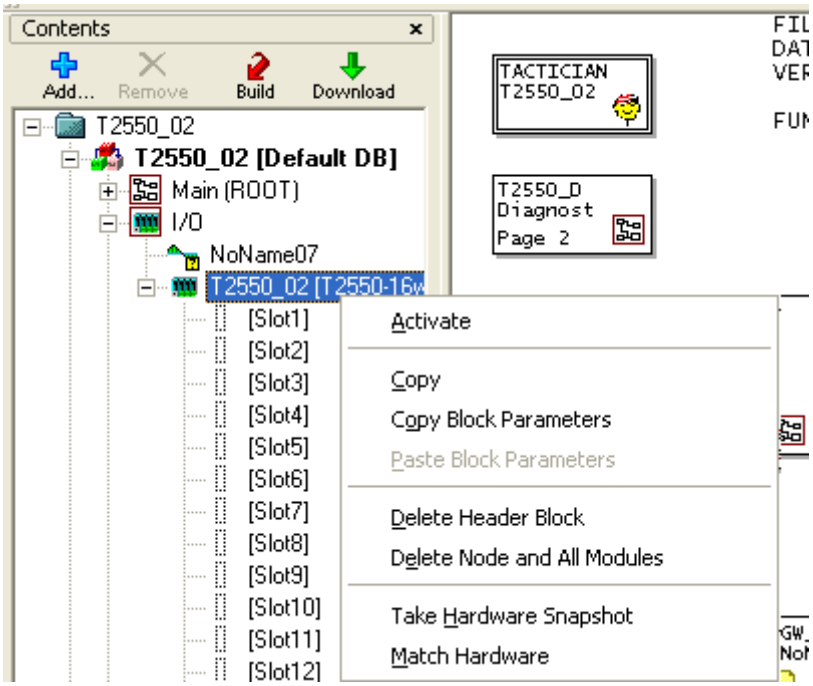
4.3 INSTRUMENT FILE COMMANDS



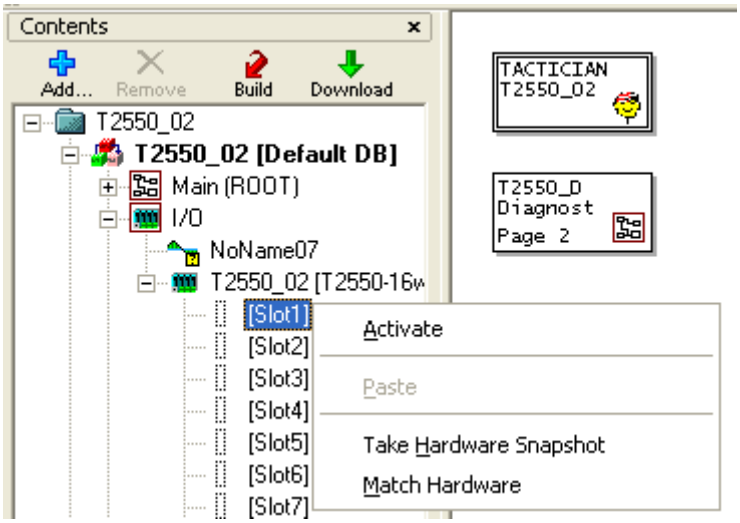
4.4 GENERAL I/O COMMANDS



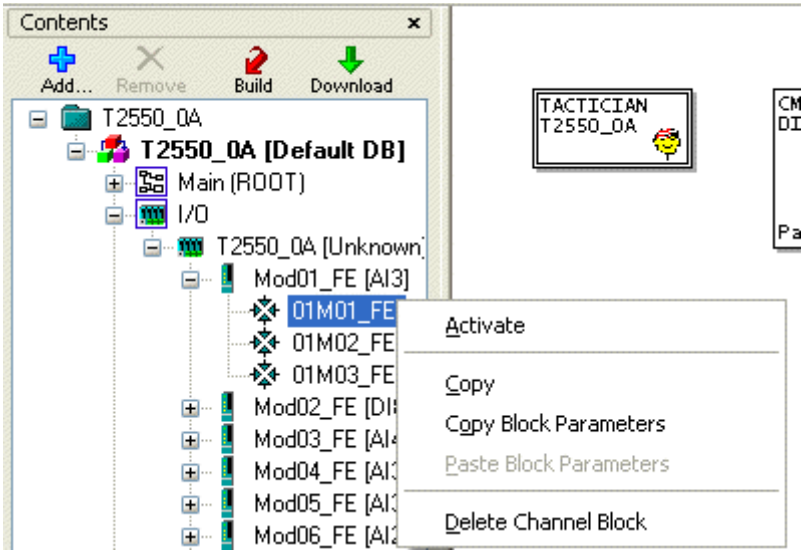
4.5 I/O INSTRUMENT COMMANDS



4.6 I/O MODULE COMMANDS



4.7 I/O CHANNEL COMMANDS



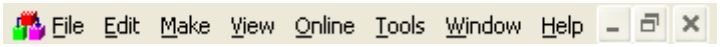
5 MENUS

The Menus can be divided into command and information areas.

5.1 MENU BAR

The Menu Bar is a special toolbar that contains pulldown commands. Each pulldown displays a further list of commands.

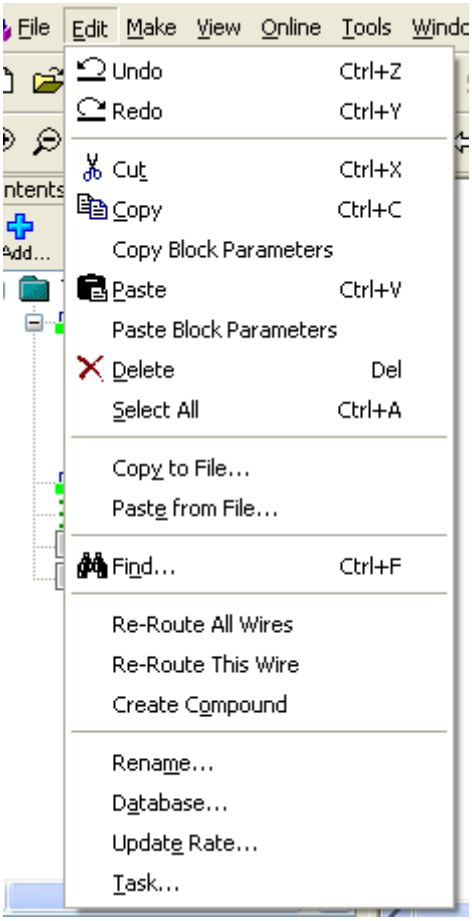
The Menu bar contains the following items,



5.1.1 Edit menu

The Edit menu enables the manipulation of an individual LINtools application file using the following commands. Each command in the figure below can be clicked on to reveal the relevant help topic.

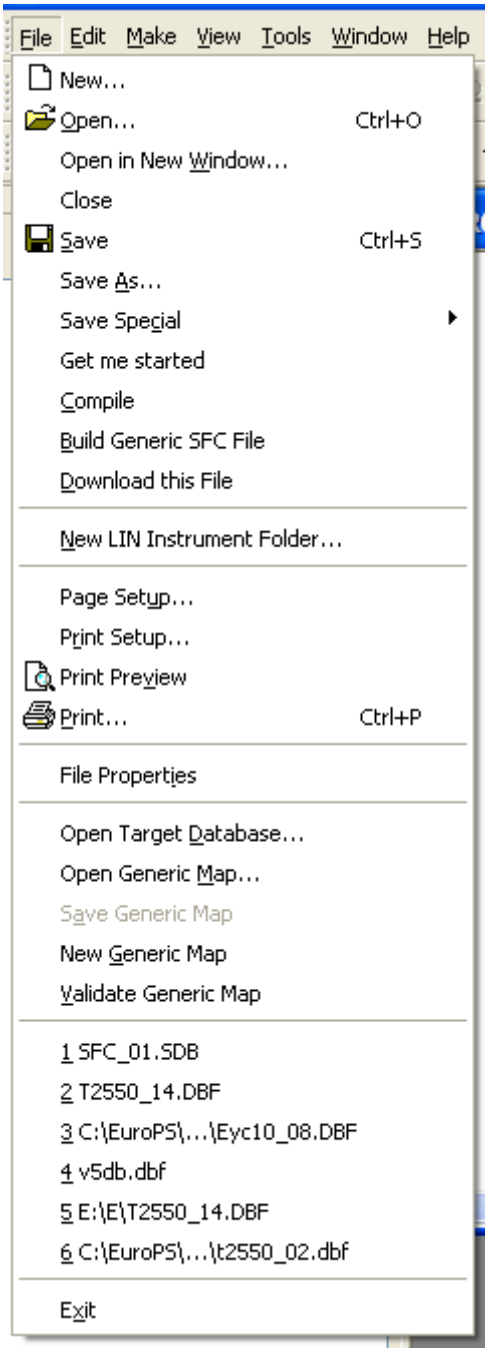
Note: The commands that appear below are file specific, i.e. the Build Generic SFC File command is available only when editing an SFC.



5.1.2 File menu

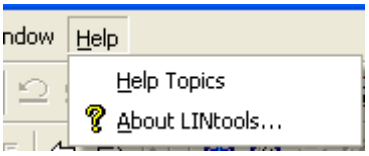
The File menu contains the items displayed below. Clicking on any item opens a topic describing the use of that item.

The menu is context sensitive. For example, the 'Build Generic SFC File' command appears only when editing a sequential function chart (SFC).



5.1.3 Help menu

The Help menu enables the used to display LINtools application details using the following commands.

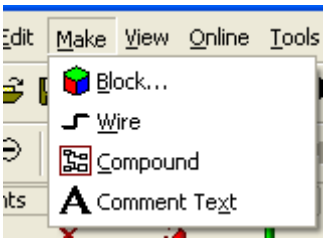


5.1.4 Make menu

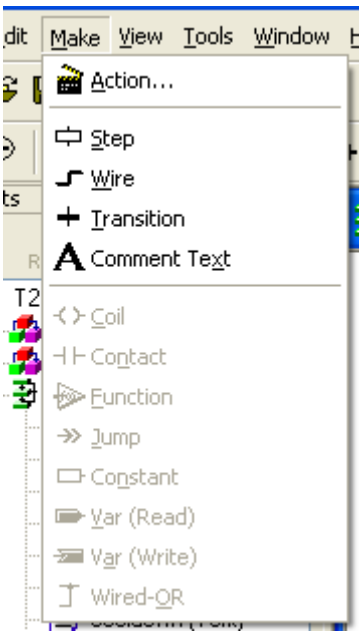
The Make menu is a list of strategy configuration commands. This provides immediate access to file specific configuration commands.

Note The available commands are file specific.

FUNCTION BLOCK DIAGRAM

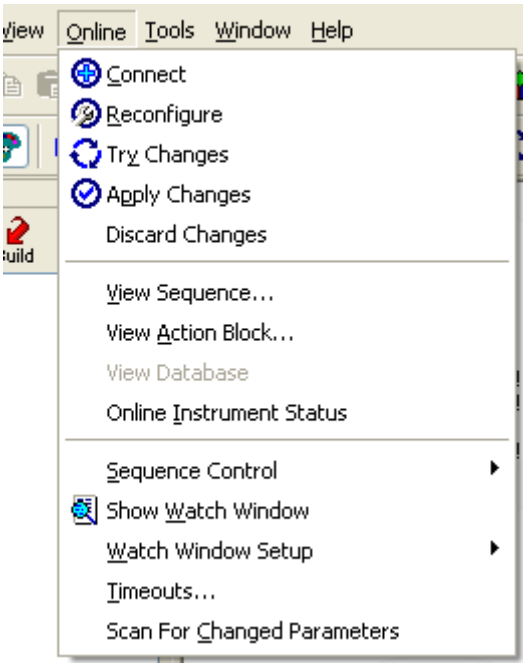


SEQUENTIAL BLOCK DIAGRAM



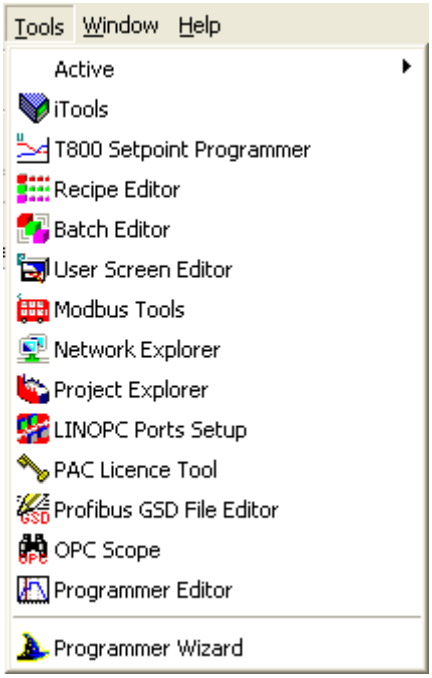
5.1.5 Online menu

The Online menu is a list of commands that provide control and management when the instrument is on-line.



5.1.6 Tools menu

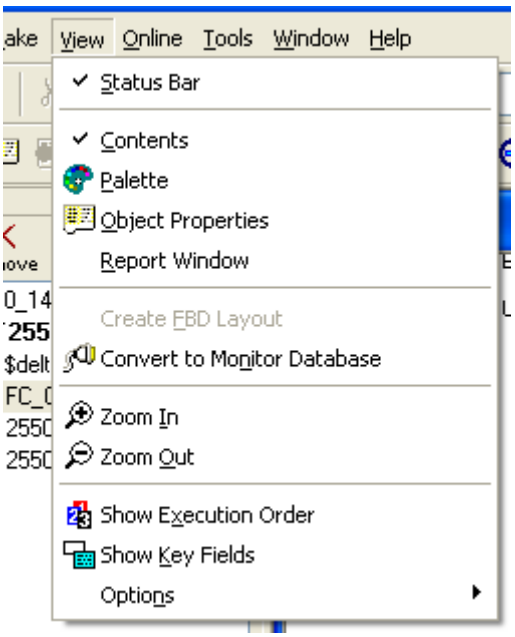
The Tools menu is a list of applications that can be launched from LINtools providing immediate access to the tools used to configure communications protocols and to create a complete instrument strategy.



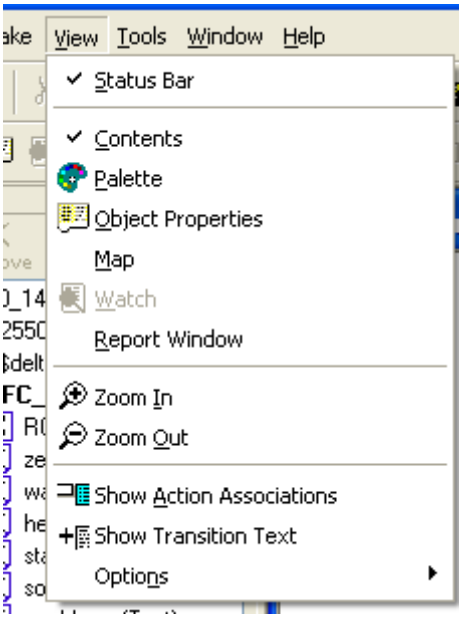
5.1.7 View menu

The View menu is a list of commands that control the appearance of LINtools application.

FUNCTION BLOCK DIAGRAM



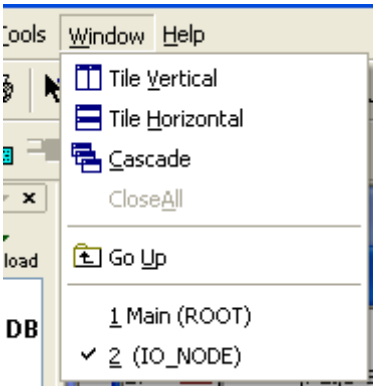
SEQUENTIAL BLOCK DIAGRAM



5.1.8 Window menu

The Window menu is a list of display styles used to arrange the currently-open sub-windows.

It also includes the 'Go Up' command which can be used to open the containing file of the current sub-window.



5.2 TOOLBARS

The Toolbars contain buttons that enable quick access to a number of Edit and View commands.

- File bar

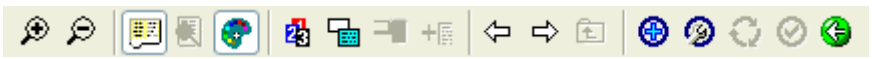
The File toolbar hosts icon buttons that enable quick access to a limited number of File commands.

- Make bar

The Make toolbar hosts icon buttons that enable quick access to a limited number of file specific configuration commands.

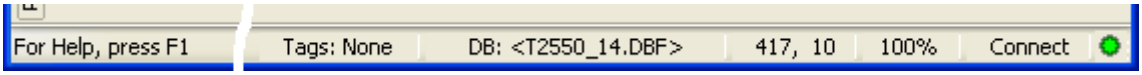
- View bar

The View toolbar hosts icon buttons that enable quick access to a limited number of LINtools application commands.



5.3 STATUS BAR

The Status Bar displays specific Editor information. It is displayed by default and is located along the foot of the screen.



6 LIN DATABASES

A **LIN Database** groups data into blocks of related data to form a control strategy for a particular application. A function block can represent an input, an output, a controller, and so on. The LIN configuration tool (LINtools Engineering Studio) and display packages (i.e. User Screen Editor) recognise different types of function block, and handle them appropriately.

A LIN Database (.dbf), also known as a 'strategy', is a database that runs in a LIN instrument providing control, data and rules. It is loaded by the LIN Instrument at runtime (typically on power up) and provides the instrument software the ability to control and monitor signals from sensors in the plant/system, (e.g. an industrial plant), and then output the signals back to actuators.

The cycle of signal input, signal processing, and signal output to the entity is repeated continuously while the database is run in the instrument using task priorities 1-4.

More than one LIN instrument can be involved in controlling a single entity, but only one LIN Database can run in a single LIN instrument at any one time.

A LIN Database can be configured as a standard single layer LIN Database or blended multi-layer LIN Database, operating in conjunction with one or more LIN Sequences (SFCs) running in the LIN instrument. It can also make use of LIN Actions (Structured Text ST/Ladder Logic)) stored in action files in the LIN instrument.

In LINtools, a LIN Database is represented and configured graphically as an arrangement of inter connected LIN function blocks, where source to destination links, are represented as wires between blocks. LIN functions blocks are picked from a library pallette and placed as appropriate in the workspace area of the LINtools Engineering Studio.

Data is passed over the network, from node to node as required by ‘caching’ function blocks as appropriate.

6.1 LIN MONITOR DATABASES

A **LIN Monitor Database** (.udm) is a LIN Database consisting entirely of cached blocks. It is constructed only for the purpose of viewing any collection of cached function blocks from any nodes on the (single) network, so any wiring and local blocks are irrelevant in a Monitor Database. However, what is relevant is the origin of the cached function blocks in the Monitor Database, i.e. from which LIN Database does this block come. The Computer operates as the LIN peer node, and cache blocks from any other nodes on the network.

Note: Use the Connect feature to inspect a collection of local blocks in a specific LIN instrument, and Online Reconfiguration to reconfigure the Strategy.

In order to create and eventually edit a LIN Monitor Database, either

- Open the required LIN Instrument Database using the LINTools Monitor program, or
- Select **View** > **Convert to Monitor Database**, if the required LIN Instrument Database is already open.

Both methods automatically create a LIN Monitor Database, including a **PC_MON** Header block usually with a fixed name, just for identification purposes. This LIN Database can then be edited to remove or add further cached blocks, and finally saved in this form, as a .udm file.

A LIN Monitor Database works with one or more LIN Monitor Sequences in the LIN instrument.

6.1.1 Monitoring a Database - Overview

Note: This assumes that Ethernet (ELIN) communications is being used.

The Monitor Database should be used to monitor only such function blocks as are selected from more than one remote database. For single node monitoring the Connect facility should be used.

To monitor a remotely running LIN Database have a 'copy' of the remote database must be loaded locally in the computer, with cached function blocks in place of the originals. This Monitor Database need not duplicate every function block in the original - just those to be monitored.

A Monitor Database is different from a standard database in that it includes a special monitor header block in place of a standard header block. Also, any local function blocks present in the Monitor Database do not update.

6.1.2 Creating a Monitor Database

A Monitor Database can be created in two main ways:

- From scratch, using the Database Monitor window, which is very much like the regular database configuration window (but without the ability to wire up function blocks).

Note: Apart from naming them correctly and assigning them to the remote database if necessary - you can leave all block fields in a monitor database totally unconfigured, with just their default values. This is because parameter data is obtained from the remote database, not the local one.

- From an existing LIN Database, by loading a remotely-running database to the Database Monitor window, where LINtools automatically converts it to a Monitor Database. Once converted, the Monitor Database can have function blocks added or removed if required, using the window's facilities. If required for future use, the Monitor Database can be saved under a different name from the original regular database.

Note: In a converted Monitor Database, existing wiring appears in the function block diagram, but is completely non-functional.

6.1.3 Interacting with the remote database

Once the Monitor Database is configured, and communicating with the remote node(s), fields in the remote database(s) can be inspected and edited via the Monitor Database.

A Watch Window can be created in the Monitor window. This allows a group of selected Database fields to be dynamically monitored and edited.

6.2 DATABASE CONFIGURATION

Databases are created and/or edited in the database editor window. The Editor can be divided in to a number of separate windows:

Function Block Diagram sub-window

Contents pane

Function block template palette

Object properties pane

Message bar

Note: Each of the panes above can be shown/hidden using the corresponding View menu command.

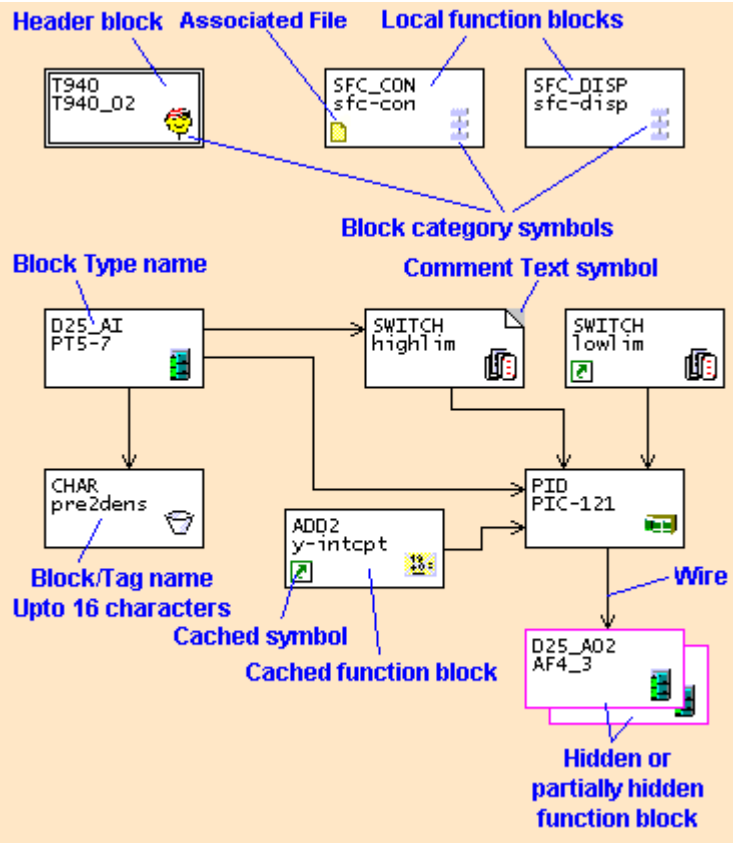
To access the Contents pane:

If the **Contents** pane is not visible, in the **View** menu, click **Contents**.

- You can cycle through the open sub-windows using the editor's Back and Forward toolbuttons.

6.2.1 Function Block Diagram (FBD) - example

An example LIN Database FBD:



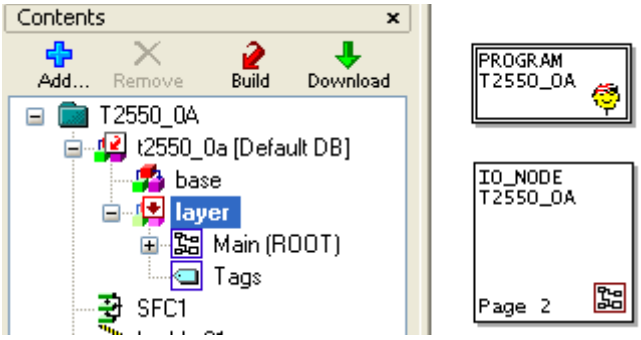
6.2.2 A Blended Database

A Blended Database (or Strategy) is a Read Only file that is the result of a Build command on a selected LIN Node containing layered LIN Database files (.dbf) and Auto generated layers (.ujc).

The Build command combines the Default DBF with any number of *.dbf and *.ujc files in a specified order as shown in the buildlst.ubl, but configured by ordering each layered LIN Database file starting immediately below the base.dbf.

- During a *Build* any PROGRAM and Cached Function Blocks are removed but all other blocks are blended into a single LIN Database.

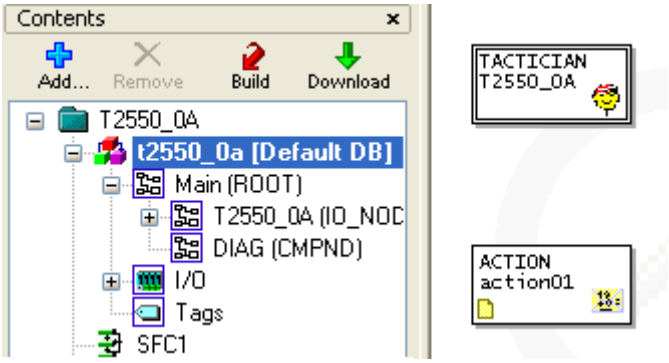
Note: The DefaultDBF and Auto generated layers are all Read Only files and do NOT require a graphics file, but each layer MUST contain a PROGRAM (Header) block with the same name as the DefaultDBF.



6.2.3 A Standard Database

A Standard Database is a Read/Write file that uses a only a single set of LIN Database files (.dbf, .dtf, and, .grf LIN instrument files).

- This is also considered as a Strategy.



6.3 CONFIGURING A LIN DATABASE - OVERVIEW

The following stages occasionally overlap, and their order can be varied, especially when an existing database is being edited. Function blocks, wiring, and parameter values can be added, modified, or deleted at any time using the LINtools toolkit.

1. Start the **New LINtools file** wizard to display a list of supported LINtools file types, either
   - Select **File** > **New** command
   - Press New LINtools file toolbutton.
2. For a new LIN Database, select New LIN Database radio button and complete each section of the Wizard.

Note: Additional information is displayed on the right of the dialog if the Help checkbox is set  .

3. Now place function blocks on the editor worksheet, in a FBD sub-window - e.g. the default '**Main**' diagram.

i. Place the required header block.
ii. Place the required function blocks, Module blocks, Diagnostic blocks and I/O blocks etc..
iii. Configure each function block with a Name and DBase. Edit any other Block fields, and add Comment tab Text to a function block, as required.
iv. Then, connect the function blocks by wires.
v. If the FBD is becoming too large for the default size worksheet, the worksheet size can be increased by converting to a large worksheet.
vi. During configuration multiple connections to a block are displayed as a single wire.
vii. Create and configure compounds as required.
viii. If the system contains Tactician instruments use the I/O table to configure the required I/O.
ix. Next add textual comments to the FBD as required.

4. Finally, save the Database to disk.

Note: Most instructions are also used when performing Online Reconfiguration. Added and deleted blocks and wires are graphically indicated

6.4 STANDARD TO BLENDED DATABASE CONFIGURATION

If changing to a Blended Database configuration, either

- the build list (build.ubl.old) is converted (to buildlst.ubl) and restored, re-establishing the Blended Database configuration.
- if no backup (build.ubl.old) exists, a new build list (buildlst.ubl) is created with the Default DBF installed as the base layer.

6.5 BLENDED TO STANDARD DATABASE CONFIGURATION

If changing to a Standard Database configuration, the layers of the existing Blended Database are separated and moved as individual files to the **Unused Files** folder. The Default DBF converts to a Standard Read/Write Database file (.dbf).

Note: A backup of the Blended Database's configuration is automatically created by converting and renaming the buildlst.ubl to build.ubl.old. Therefore this will not be recognized as a Build file.

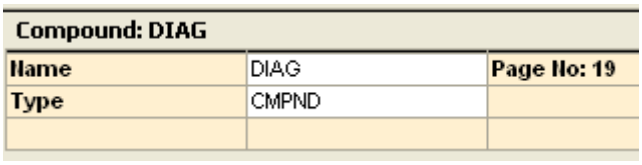
6.6 FUNCTION BLOCK DIAGRAM OBJECT PROPERTIES PANE

This Object properties pane shows all aspects in the instrument, and the field values in the function block highlighted in the main window.

If this pane is not visible, select **View** > **Object Properties**. The ▾/▴ symbol can be used to minimise or maximise the Pane if more than one pane is docked in the same area of the application, and the ✕ can be used to hide this pane.

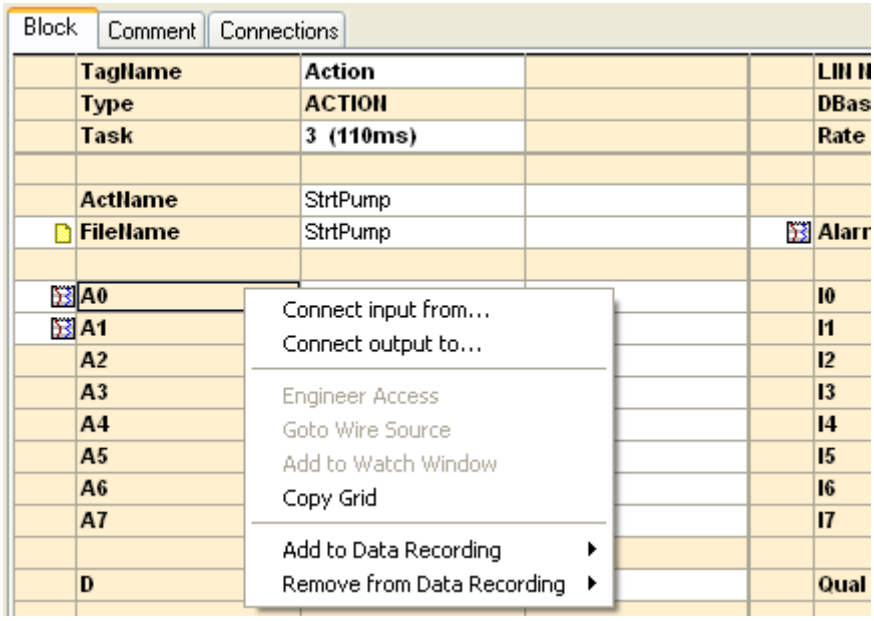
6.6.1 Compound tab

This is the **Name** and **Type** of the compound highlighted in the main window can be edited. This is not a tab but is displayed in this position when a compound is highlighted in the main window.



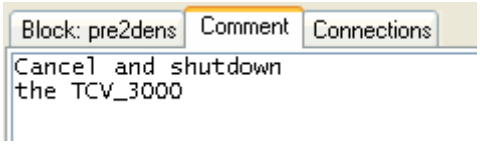
6.6.2 Block tab

Right clicking on a field causes a drop down menu to appear, as shown below, allowing function block fields to be inspected & manipulated and fields to be added to the Watch Window. The drop down menu also allows the user to trace the source of any inputs wired to fields in the relevant function block.



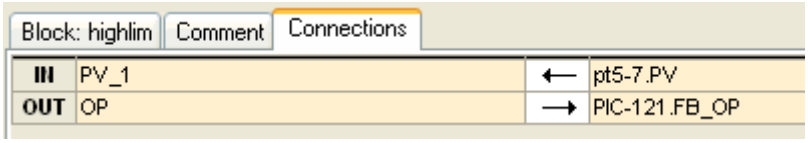
6.6.3 Comment tab

This area allows textual comments to be added to the function block. Once added, such comments can be freely edited.



6.6.4 Connections tab

This is where you inspect the connection(s) in the wire highlighted in the main window.



Note: You cannot add wiring to a monitor database, or configure Loopback.

6.6.5 Database file properties

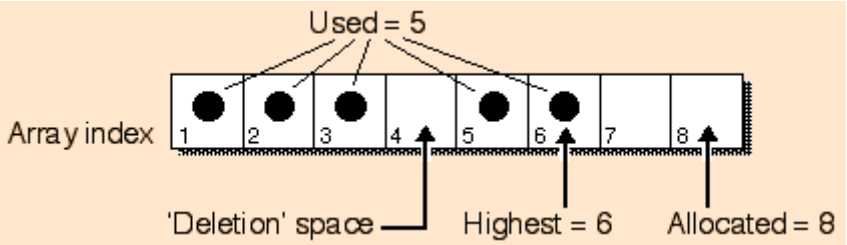
Sequence Properties dialog

This read-only page summarises the resources used by or allocated to the current LIN Database. The page is accessed by clicking on 'Loaded File Properties' in the file menu.

6.6.6 Resources

Resources are the Function blocks, Function block templates, libraries, external databases, etc., that are used by or allocated to the Database. In the Database **Properties** page, each resource has three numbers applying to it - **Used**, **Highest**, and **Max**.

The figure shows schematically what *Used* and *Highest* mean. In the image below, five items of the resource, represented by black circles, are located in an array of eight indexed memory cells.



- **Used** - This is the quantity of the resource (e.g. number of blocks) that is actually being used by the database. In the figure, *Used* = 5.
- **Highest** - This is the highest index number in the allocated array that is occupied by a resource item. *Highest* = 6 in the figure. *Highest* may be greater than *Used* if any gaps have been left by deletions during configuration, e.g. in the figure, the fourth item created during configuration was subsequently deleted.
- **Max** - This is the number of memory cells allocated by the Editor to the resource, and is fixed at the displayed quantity (not shown in the diagram). **Max** takes no account of target instrument memory limitations.

DATABASE FREE SPACE

This is the percentage of the available memory left to store the Database.

TEMPLATE FREE SPACE

This is the percentage of the available memory left to store the templates used by the Database.

DATABASE NAME

This is the name by which this Database will be known across the LIN communications. It is also the name of the header block. Use this name when cacheing function blocks from this Database into other instruments.

HEADER TYPE

This is the type of LIN instrument for which this Database has been configured. It is the name of the template used to create the first function block - header block - in the Database.

TARGET

This is the name of the directory from which the templates used in this database were selected during configuration. This indicates the target LIN instrument type and version for which the Database has been constructed.

PATH

Shows the location of the database within the host computer.

ENABLE ALARM SUPPRESSION

For relevant instruments, this allows the user to enable/disable alarm suppression.

7 TAG CONFIGURATION

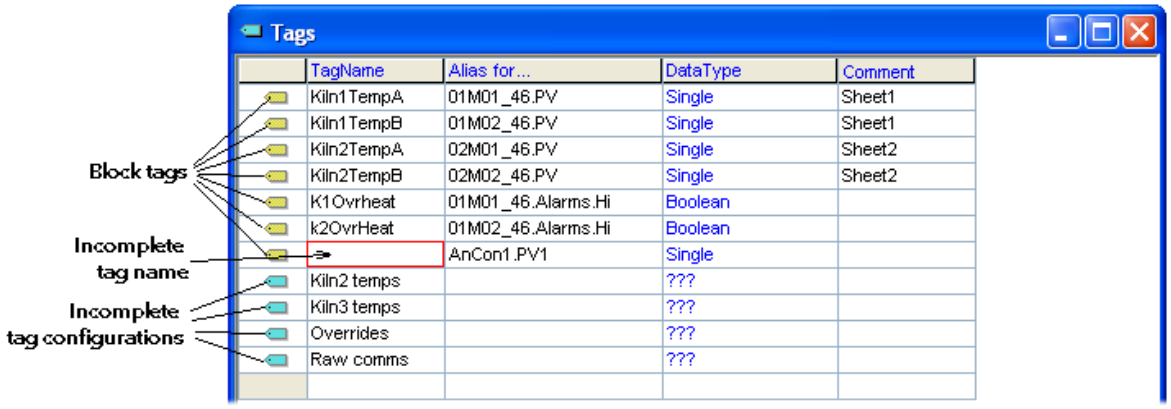
Tag configuration is the facility to create and edit either up to an 8 or 16 character name, as configured in the **Settings** dialog, to represent

- any *Common Tag*, including
- an individual data field or subfield, (*Field Tag*)
- a generic I/O point, (*I/O Tag*)
- a LIN Function block, (*Block Tag*)
- or a LIN Control Module Tag, (*Special Tag*).

The configuration is achieved via the 'Tags' sub-window.

Use the Tag table

An example Tag configuration:



7.1 CONFIGURE TAGS - OVERVIEW

To configure the Tags, follow these stages.

Stages occasionally overlap, and their order can be varied, especially when existing Tags are being edited. TagNames and Aliases can be added, modified or deleted at any time using the LINtools toolkit.

1. For a new Tag, open the Tag configuration tool to display the Tag table sub-window.
   - Alternatively, with the Content list tab in Tree view, click the '+' until the Tag sub-window is displayed.

Notes:
1. If editing an existing instrument some Tag table cells may already be complete. Individual Tags can be deleted using the Cut command, whereas all the Tags can be deleted using the Clear ALL Entries command.
2. Pressing the PC's <Ctrl>+<F6> keys cycles the currently open sub windows.

2. In the Tag table sub-window, configure the required Tags within the constraints configured in the Setting dialog.

i. Enter a Tag Name in the first available cell in the TagName column. This is an assumed name for the item selected in the Alias For… column.

ii. Select an Alias for the TagName. Double-click the corresponding Alias For… cell and from the drop down list select what the TagName is an alias for, i.e. which block, field or subfield it represents.

iii. If required, enter relevant information in the corresponding Comments cell.

3. Finally, save the Tag configuration.

8 I/O CONFIGURATION

I/O configuration is the facility to create and edit I/O block strategies running in Tactician instruments ONLY across the ELIN, providing a versatile commissioning and strategy debugging toolkit.

In LINtools, an I/O configuration is an extension of the LIN Database and is configured graphically as an arrangement of LIN function blocks interconnected by software wires.

Note: As with a LIN Database, the I/O configuration can also be reconfigured online.

LINtools can automatically create I/O blocks using the Match Hardware command, achieved by communicating with I/O system and creating the appropriate blocks in the instrument's database. It determines the number of slots available and then how many and what types of module are used. It also determines the number of channel blocks needed for each module.

It is also possible to create an I/O system manually by creating a Header block, the same number of MOD_UIO blocks as the fitted hardware modules and configuring the appropriate Module Type.

Example

If the I/O Node (e.g. T2550) is using 3 Modules (AI2 in slot 1, AO4 in slot 2, and AI2 in slot 3), LINtools must have 1 Tactician Header block and 3 MOD_UIO blocks (automatically configured with the type and the number of channel blocks required).

8.1 I/O CONFIGURATION - OVERVIEW

Note: The I/O Configurator is not available for all instruments. The I/O configuration is not displayed in Monitor mode.

The following stages occasionally overlap, and their order can be varied, especially when editing an existing I/O instrument. Function blocks (including I/O blocks), wiring, and parameter values can be added, modified, or deleted at any time using the LINtools toolkit.

Note: If the I/O Node already exists on the network this process can be avoided by using the Match Hardware command. This command attaches to the target instrument automatically adding blocks that match the hardware fitted exactly.

1. For a new I/O configuration, open the I/O configurator in the **Contents tab** to display the I/O table sub-window.

   Alternatively, with the **Content tab** in tree view, press the '+' until the I/O sub-window is displayed.

Notes:

1. If editing an existing I/O instrument the I/O table cells and Content list I/O slots may already be complete, but can be deleted using the appropriate Delete command from the context-sensitive menu.
2. Pressing the PC's <Ctrl-F6> keys cycles the open sub-windows.

2. In the I/O table sub-window, configure the I/O Node.
   i. Enter the I/O Node Name in the appropriate cell.
   ii. Double-click the I/O Node Type cell and select the required I/O Node Type from the drop-down list.
3. Then, specify each of the I/O modules required.

   As each of the following steps is completed, Module icons appear in the Contents list. This indicates that an I/O Slot compound containing a MOD_UIO block has been successfully created.

   i. Enter the Module Names in the appropriate slot cells.
   ii. Double-click the corresponding Type cell and select the required Module Type from the drop-down list.

iii. If the I/O hardware is present on the network, use the Match Hardware command to automatically complete the I/O table.

Note: Existing Module configurations can be moved or copied (using the Ctrl key) by dragging and dropping a required Module in to a different slot.

4. Now, configure the required Module Channels.

Note: I/O Channel blocks do not have to remain in the Compounds in which they were created. Move the I/O Channel block to an appropriate location in the database to simplify the wiring. The compounds are used to simplify the identification of the Modules when first created.

As each of the following steps is completed, '+' icons appear in the **Contents tab**. This indicates that the specified number of Channel blocks have been successfully created in the I/O Slot compound.

i. Double-click a configured Module Slot number to reveal a list of available channels.

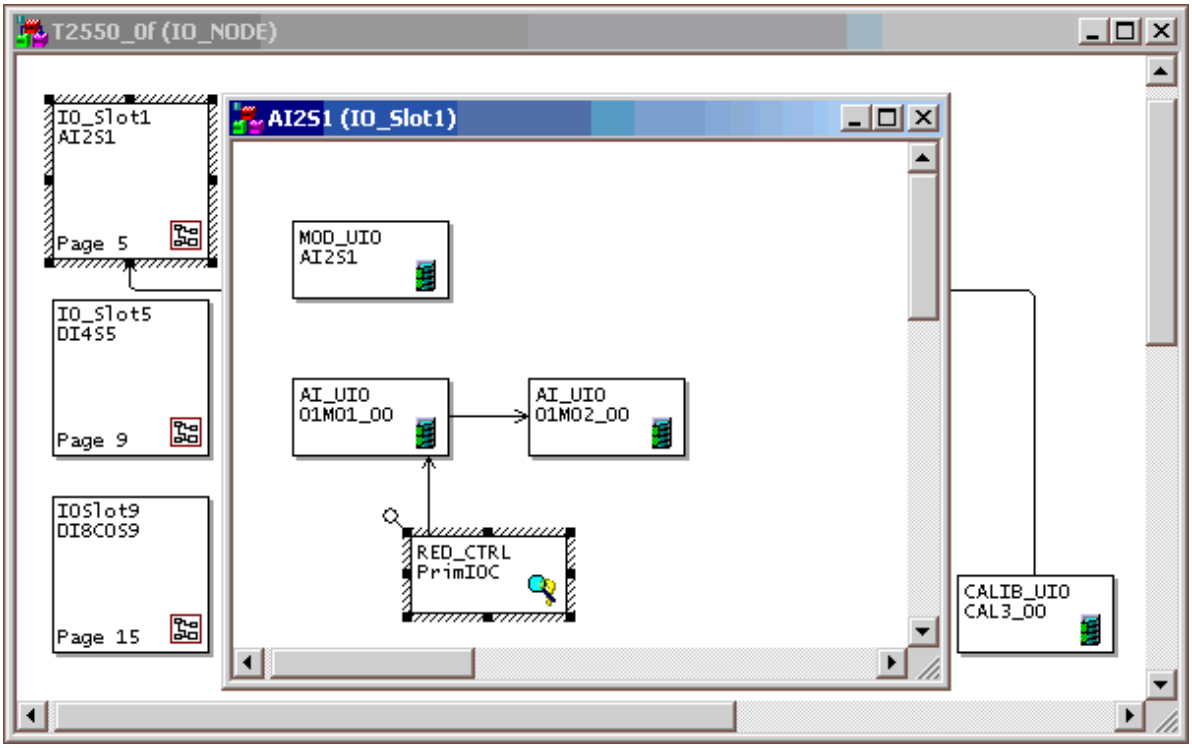
ii. Enter the Channel Names in the appropriate channel cells.

- Right click to display a context-sensitive menu and select Autocreate Channel blocks to create individual Channel blocks or,
- Right-click the Module Slot number to display a context-sensitive menu and select Autocreate Channel blocks to create all Channel blocks.

Note: To move an existing I/O Channel to another location, change the MOD_UIO blocks SiteNo or Channel field as required.

5. Finally, save the I/O Configuration to disk.

An example I/O LIN Database FBD:



9 LIN SEQUENCES

A **LIN Sequence** is a program that runs in a instrument, in conjunction with a Database. More than one Sequence can run simultaneously in the instrument.

A Sequence interacts with its associated Database by writing new values to specified Database fields, in response to changes in the values of other specified Database fields.

Note: In multi-tasking instruments, e.g. T2550 and T940(X), the SFC_CON block and any blocks interfacing this block must always operate on User Task 4.

A Sequence is used when the process being controlled by the Database can adopt several distinct **states** - e.g. 'Starting Up', 'Full Running', 'Shutting Down', etc.

The purpose of a Sequence is to determine:

- The initial state(s) adopted by the process at start-up.
- The conditions triggering state-changes (events).
- The new state(s) adopted when changes are triggered.
- The way the Database controls the process in each of the possible states.

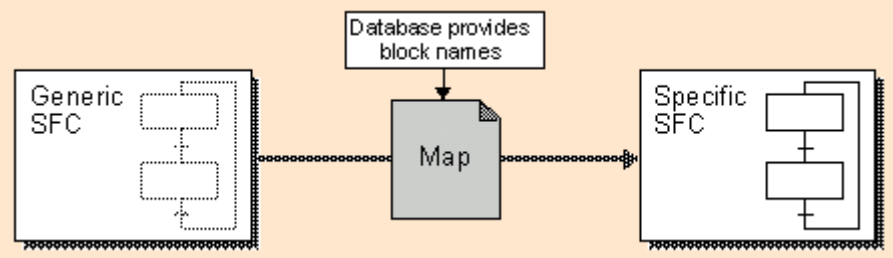
9.1 SPECIFIC & GENERIC SEQUENCES

There are two types of LIN Sequence - Specific and Generic:

- A Specific SFC is a 'working' SFC, designed to run with a particular control Database. It contains Structured Text statements and expressions that refer to particular points in the specified control Database.
- A Generic SFC is a 'template' for a specific SFC. It does not refer to specifics, but instead contains generic names that can subsequently be made to represent real Database function blocks or points by creating a map file that associates each generic name with a particular function block or point in a specified control Database.

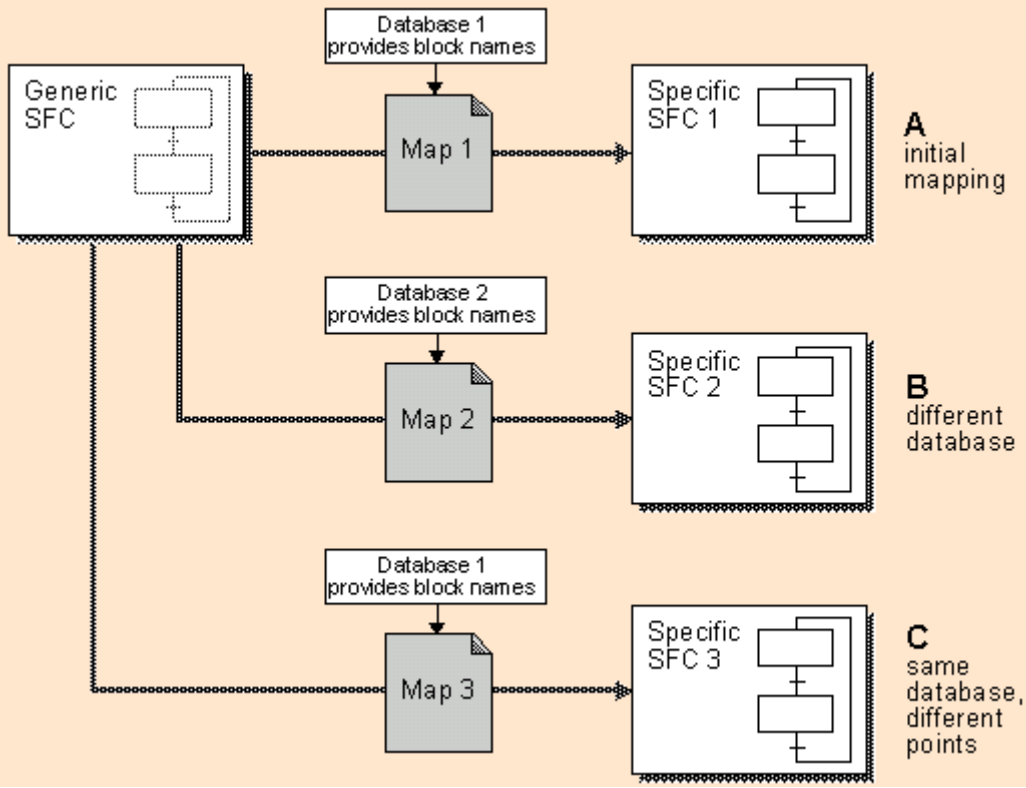
In other words, combining a map file with a Generic SFC creates a Specific SFC - i.e. an instance of the Generic template SFC:

9.1.1 Case A



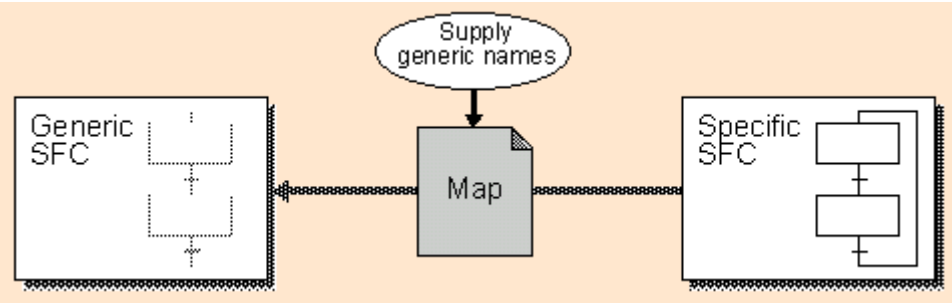
By combining with different maps, the same Generic SFC can generate any number of different (but identically structured) Specific SFCs. A new mapping can combine the Generic SFC with a new control Database (case B in the figure), or with different points in the same Database (case C):

9.1.2 Case B



A Generic SFC can also be created from a Specific SFC. This is useful when there is a working Specific SFC and it is required to generate a set of similar SFCs to run with related control Databases:

9.1.3 Case C



Note: An advantage with Generic SFCs is that generic names are up to 16 characters long and can be very descriptive compared to the often nondescript specific block/field/bit Database names. E.g. the generic name 'WaterLevelLoFlag' could map to the real point 'DG_CON88.W Field3.Bit9'. This clarity is especially useful during SFC development, or modification to run with other Databases.

9.1.4 Specificity of generic name mapping

Generic names can be mapped to database names in three ways, according to how specific the mapping is:

- **Block.field.subfield mapping -** The generic name maps to a complete block.field.subfield LIN Database point, e.g. PressureAlarm maps to PID_01.Alarms.HighAbs.
- **Block.field mapping -** The generic name maps to a block and field only, leaving the subfield (if any) unspecified, e.g. TempHiFlag maps to DGOUT_01.Out
  The rest of the LIN Database point (i.e. the particular bit of the **Out** field) must be specified in the Generic Sequence Action or Transition text, e.g. as **TempHiFlag.**Bit3.

- **Block mapping -** The generic name maps to a block TagName only, leaving the field and subfield (if any) unspecified, e.g. KilnController maps to TIC015
  The rest of the LIN Database point is specified in the Generic Sequence, e.g. as **KilnController.**OP.

9.2 SEQUENCE EDITOR - OVERVIEW

The links on this page display further topics describing the various sequence editor features.

This Editor can be divided in to a number of separate panes:

Sequential Function Chart sub-window

Note: Each of the following panes can be individually manoeuvred in the application window, e.g. the Template Palette can be docked below the Contents Pane.

Contents pane

Template Palette

Object Properties pane

Compile Results pane

Message bar

Note: Each of the panes above can be shown/hidden using the corresponding View menu command.

9.2.1 Configuring a LIN Sequence

Note: Try to ensure sufficient comments are made throughout the configuration of a LIN Sequence. The comments will enable a better understanding of the configuration, should problems arise at a later date.

The following stages occasionally overlap, and their order can be varied, especially when you are editing an existing Sequence.

Note: In multi-tasking instruments, e.g. T2550, T940(X), the SFC_CON block (the interface between the Database and the Sequence) and any blocks interfacing with a Sequence must always operate on User Task 4.

1. For a new LIN Sequence, first create a LIN Sequence file.

Note: The 'Get me started' wizard can assist with creating new LINtools files.

2. Open the LIN Sequence file in the LIN Sequence Editor window.
3. For a Generic Sequence, open a Map file and configure it as far as possible. Validate the map to enable the generic name PickList accessed via Variable, which will be needed in steps **7** and **9**.

Note: Pressing the computer's <F9> key displays the Variable Picklist. A filtered Picklist may be expanded to show all available variables by pressing the <F8> key.

4. Now, place Steps on the editor worksheet in the ROOT chart.
5. Then connect the Steps by wires and Transitions.
   - If the Steps are becoming too large for the default size worksheet, the worksheet size can be increased by converting to a large worksheet.
6. Configure each Step with a Step name and initial state. Add Comment Tab Text to the Step, if required.
7. Next, create Structured Text, Sequential Function Chart, and Ladder Diagram Actions (as required) that will run in the Steps.

8. Then associate the appropriate Actions with each Step in the LIN Sequence, via Action Qualifiers.
9. For each Transition, write a Transition expression. Add Comment Tab Text to the Transition, if required.
10. If required, add textual comments to the ROOT Chart, as required.
11. After adding textual comments access any Chart Actions and configure them as for the ROOT chart.
    - Then compile the LIN Sequence to check the validity of your Sequence Actions and Transitions. Correct all errors.
    - Finally save the Sequence to disk.

9.3 MONITORING A SEQUENCE

9.3.1 The Sequence Action monitor window

A Sequence running in a remote instrument can be displayed locally as a dynamically active Sequential Function Chart (SFC), in the **Sequence Action Monitor** window. After accessing the Sequence Action Monitor window, this display can be used to monitor and control the remote Sequence.

Specifically, from the Sequence Action Monitor window you can:

1. See the currently-active Step(s) highlighted
2. Hold Steps in their active state
3. Force Transitions to act as if TRUE or FALSE
4. Reset, Stop, Hold, and Run the remote Sequence
5. Find and inspect Actions, associations, and Steps
6. Find and inspect any block or field in the associated LIN Database
7. Monitor a selection of the associated Database fields in a Watch window
8. Monitor a Ladder Diagram Sequence Action
9. Monitor a Ladder Diagram Transition
10. Monitor a LIN Action

Note: For monitoring to be possible, the remote LIN Database associated with the LIN Sequence must have an SFC_CON block present and linked to the appropriate SFC_DISP block.

Local copies of the LIN Sequence and graphics files (.sdb and .sgx) are required to allow the corresponding remote Sequence to be monitored.

9.4 SEQUENCE ACTION MONITOR - OVERVIEW

The Sequence Action Monitor window is where a Sequence (and its associated Database) can be monitored and controlled. The links below jump to further topics describing the functions of the Sequence Action Monitor..

Note: Actions cannot be edited in the Sequence Monitor window,but the associated Database Monitor can easily be accessed.

This Monitor window can be divided in to a number of separate panes:

Sequential Function Chart sub-window

Note: Each of the following panes can be individually manoeuvred in the application window, e.g. the Template Palette can be docked below the Contents Pane.

Contents pane

Function block template palette

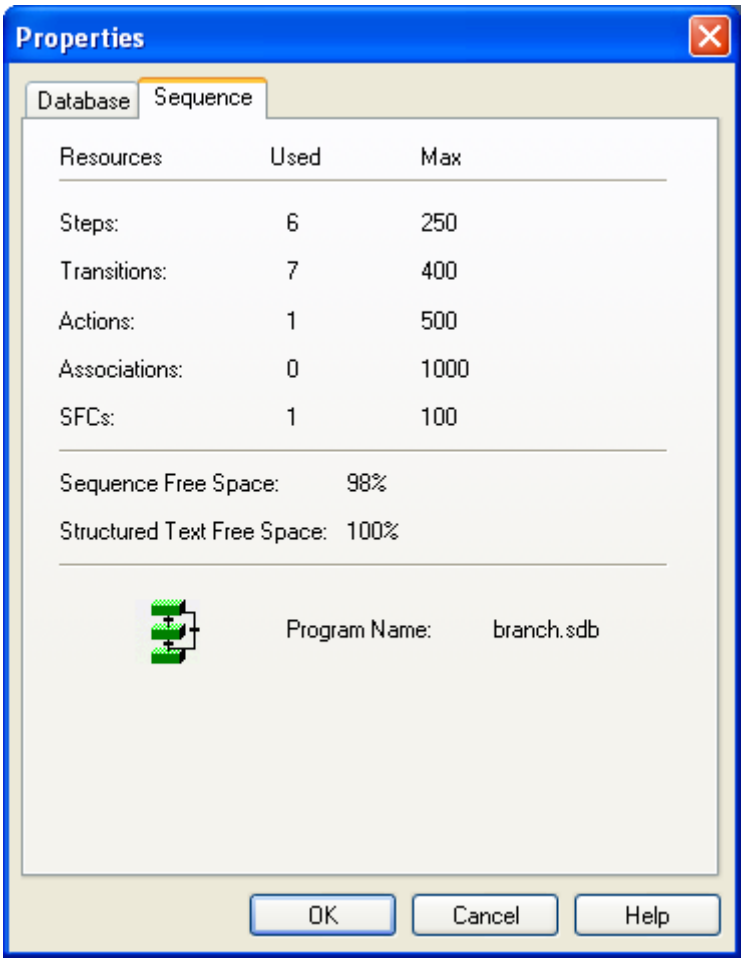
Object properties pane

Message bar

Note: Each of the panes above can be shown/hidden using the corresponding View menu command.

9.5 SEQUENCE PROPERTIES DISPLAY

This read-only page summarises the resources used by the current LIN Sequence. The display is accessed by selecting 'Loaded File Properties' in the File menu and clicking on the 'Sequence' tab.



9.5.1 Resources

Resources are the Steps, Transitions, Actions, Associations, and SFCs, that are used by the Sequence.

9.5.2 Used

This is the quantity of the resource that is being used by the LIN Sequence.

- the Steps - Instruments have a limit of either 150 or 390 total steps loaded.
- the Transitions - Instruments have a limit of either 225 or 578 total Transitions loaded.
- the Actions - Instruments have a limit of either 300 or 780 total Actions loaded.
- the Associations - Instruments have a limit of either 600 or 1560 total associations loaded.
- the SFCs - Instruments have a limit of either 50 or 130 total SFC's loaded, including the root SFC.

9.5.3 Sequence Free Space

This is the percentage of the available memory left to store the Sequence.

9.5.4 Structured Text Free Space

This is the percentage of the available memory left to store the Structured Text used by the Sequence.

9.5.5 Program Name

This is the name of the LIN Sequence program - which is the short name of the file containing the selected Sequence.

10 LIN ACTIONS

Two types of LIN Action exist:

- A Text action, i.e. a set of instructions written in Structured Text (ST), or
- A Ladder Diagram action, with graphical rungs, contacts, coils, and variables, etc.

Both types specify direct operations on the LIN Database of the running Strategy.

LIN Actions can be created in the LIN Action Editor.

LIN actions created in the action editor work in a similar way to the Actions created in Sequence Editor, but there are important differences.

10.1 DIFFERENCES BETWEEN LIN ACTIONS AND SEQUENCE ACTIONS:

- LIN Actions are controlled - via Action Qualifiers - and activated directly by certain 'Action-type' function blocks in the LIN Database. They are **not** associated with Sequence Steps, are **not** accessible from the Sequence Editor, and indeed play no part in Sequences at all.
- LIN Actions can operate only on the fields provided within **one** of these action-type function blocks, whereas a Sequence Action can operate on any fields in the entire LIN Database.
- LIN Actions are stored in their own special user-named 'Action files' that must be downloaded to the target instrument with the other strategy database files. Sequence actions, however, are stored along with the Sequence Database itself and so do not need to be individually downloaded.

Note: The Action Editor yields files that are independent of any particular LIN Database (.dbf). The text and ladder actions can be run with any suitable Database.

10.2 ACTION CONFIGURATION

Note: A new Ladder Action can be created, but the Chart Action is not supported in this Editor.

The Action Editor is where LIN Action files are created and / or edited.

This Editor is divided in to a number of separate panes:

Text Action sub-window

Note: Each of the following panes can be individually manoeuvred in the application window e.g. the Template Palette can be docked below the Contents Pane. Panes can also be individually shown or hidden using the relevant ‘View’ menu command.

Contents pane

Template palette

Message bar

10.2.1 Action names

Action names can have up to eight characters, including only a-z, A-Z, 0-9, _. Action names must not start with a numeral.

10.2.2 Action types

There are three types of Action that can be associated with a Sequence Step:

- **Chart action -** This action is itself a LIN Sequence, represented as a Sequential Function Chart (SFC).
- **Text action -** This action consists purely of Structured Text statements.
- **Ladder Diagram action -** This action is represented pictorially as a Ladder Diagram, with rungs, contacts, coils, and variables, etc.

Note: LIN Actions are unconnected with LIN Sequences. Only Structured Text and Ladder Diagram types can be created for LIN Actions.

10.2.3 Action-type function blocks

An 'Action-type' LIN function block, running in a Database, can be associated with a LIN Action via an action qualifier. This allows Sequence-like Actions to operate on the Database, independently of any LIN Sequence. In this type of function block, an input triggers the operation of the LIN Action.

- Action-type blocks include the **ACTION** and **DIGACT** function blocks.

10.3 CONFIGURING A LIN ACTION

LIN Actions are created and/or edited using the Action Editor.

To create a new LIN Action:

1. Open a blank LINtools file, and create a new **Action File** type configuration.

   The **Action Editor** opens, displaying a default **Contents** pane, blank window, and a Function Block Template Palette.

- Alternatively, you can create a new Action files within the Project environment:

2. The default Action window is for configuring an Action to be run by the **ACTION** function block. To create an action for any other action-type block, double-click the required function block in the function block template palette to show a Make Action (block type) dialog.
3. Type in a name for the new Action and select the Action type required - LIN Actions can be either **text** or **ladder diagram** types. Click on **OK**. A new empty Action window opens up, showing the target block type in brackets in the title bar - e.g. **action1 (DIGACT)**.
4. Type the Structured Text (ST) for a text Action into its Action window, or build up a Ladder Diagram from the available elements.
5. Compile the Action at any time to check its validity.
6. If required, repeat steps **3** to **6** to create more Actions in the Action window.
7. When finished, save the Action(s) in the Editor.

Note: All the LIN Actions in the current configuration are saved immediately to the same file.

11 LADDER DIAGRAMS

A Ladder Diagram is a type of Action represented graphically by a column of 'rungs' - see example. Rungs are equivalent to program statements, with icons along them representing digital or analog fields, constants, and logical or arithmetic functions.

Each rung has only one 'output' or 'objective' - at its right-hand end - which is either a coil (digital field), variable (analogue field), or a 'jump' to another labelled rung.

A single rung that evaluates TRUE or FALSE can also act as a Sequence Transition.

Rungs can include any number of input elements and use any complexity of wired or explicit functions to perform the rung operation - subject only to screen space limitations

The rungs are normally executed in order from top to bottom down the ladder, but jumps can alter this.

In a ladder diagram, a '**closed**' switch or coil equates to the **TRUE** (logical '1') state and an '**open**' one equates to the **FALSE** (logical '0') state of the associated field.

11.1 LADDER DIAGRAM ELEMENTS

You click on these tools to insert elements along the rungs of your Ladder Diagram.

Editing a Ladder Diagram

11.1.1 Coil

There are four types of coil:

The **Normally-Open coil**  outputs FALSE to its associated field if the rung evaluates FALSE, and outputs TRUE otherwise.

The **Normally-Closed coil**  outputs TRUE to its associated field if the rung evaluates FALSE, and outputs FALSE otherwise.

The **Reset coil**  outputs FALSE to its associated field if the rung evaluates TRUE, and does nothing otherwise.

The **Set coil** outputs TRUE to its associated field if the rung evaluates TRUE, and does nothing otherwise.

11.1.2 Contact

There are two types of contact:

The **Normally-Open contact**  evaluates FALSE if the associated field is FALSE, and evaluates TRUE otherwise.

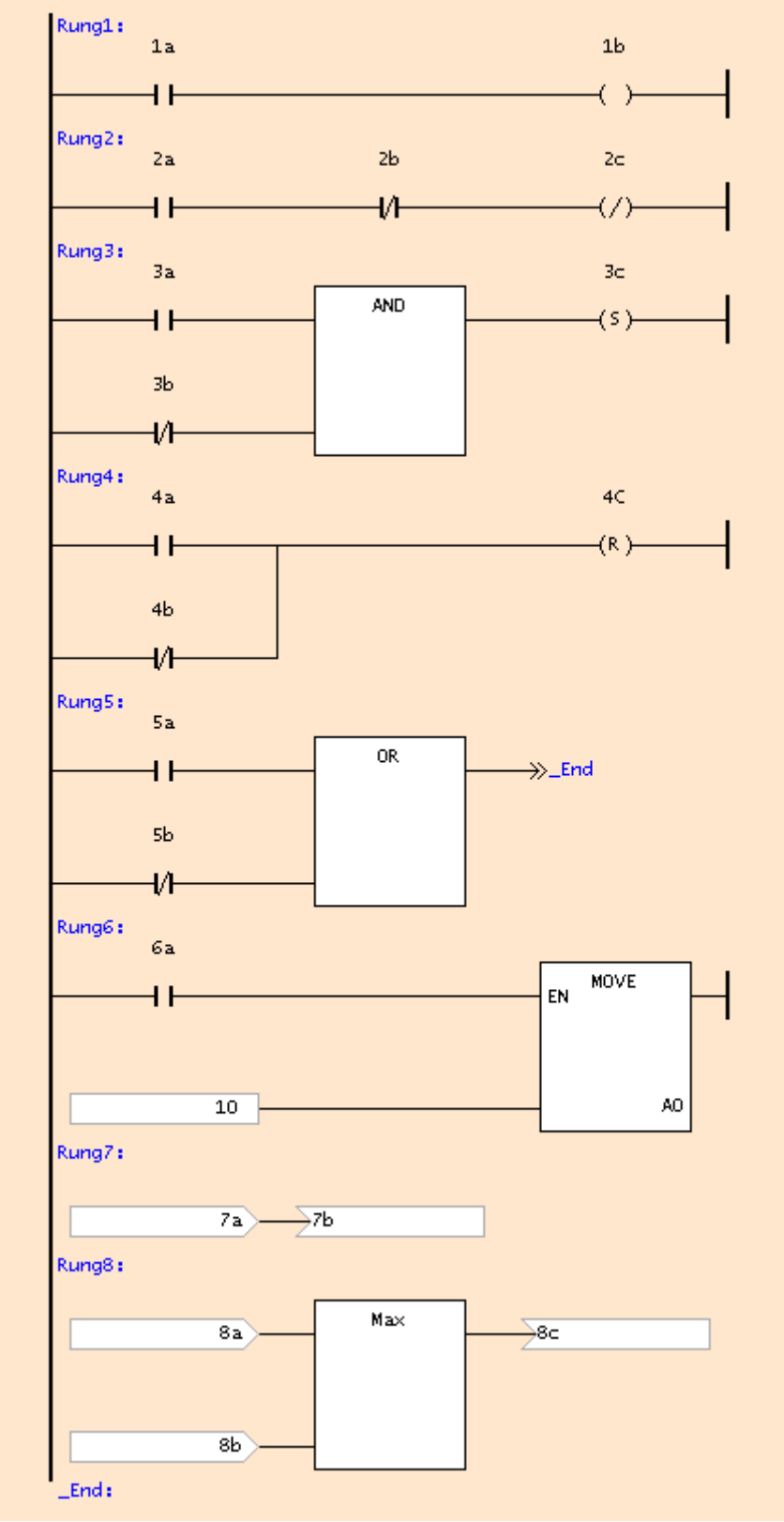
The **Normally-Closed contact**  evaluates TRUE if the associated field is FALSE, and evaluates FALSE otherwise.

11.1.3 Jump

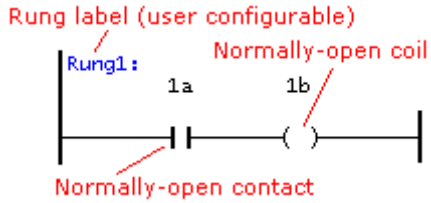
A jump causes the execution of the ladder to continue at the rung specified by the jump. Jumps can be only forwards.

11.2 LADDER DIAGRAM – EXAMPLES

11.2.1 Overview



11.2.2 Rung 1 - example

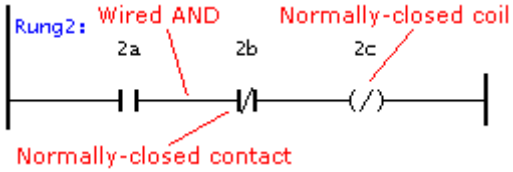


Rung 1 performs the following operation:

If digital 1a is TRUE, then digital 1b is set to TRUE, otherwise, digital 1b is set to FALSE.

Using a 'circuit' model, if switch 1a is closed - by digital 1a being TRUE - then coil 1b is powered up and closes, i.e. becomes TRUE. With switch 1a open (digital 1a FALSE), coil 1b adopts its unpowered normally-open state (= FALSE).

11.2.3 Rung 2 - example



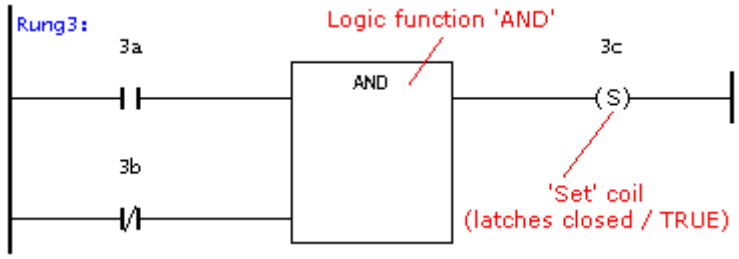
Rung 2 performs the following operation:

If digital 2a is TRUE, AND digital 2b is FALSE, then digital 2c is set to FALSE, otherwise, digital 2c is set to TRUE.

Using a 'circuit' model, if switch 2a is closed - by digital 2a being TRUE - and switch 2b is left closed - by digital 2b being FALSE - then coil 2c is powered up and opens, i.e. becomes FALSE. Otherwise, coil 2c is unpowered and adopts its normally closed (= TRUE) state.

This arrangement is an example of a '**wired AND**'. **Rung 3** shows an alternative approach.

11.2.4 Rung 3 - example



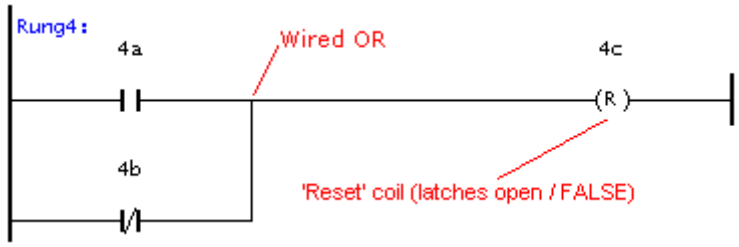
Rung 3 performs the following operation:

If digital 3a is TRUE, AND digital 3b is FALSE, then digital 3c is set to TRUE, otherwise, digital 3c is unchanged.

A 'set' coil latches closed (TRUE) when 'powered up', and does not reset (to FALSE) when power is removed.

This arrangement uses an explicit **AND function**. **Rung 2** shows an alternative approach.

11.2.5 Rung 4 - example



Rung 4 performs the following operation:

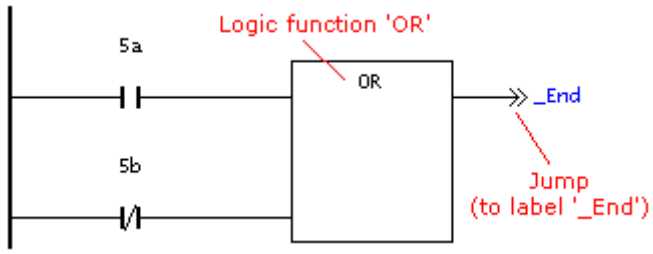
If digital 4a is TRUE, OR digital 4b is FALSE, then digital 4c is reset to FALSE,

otherwise, digital 4c is unchanged.

A 'reset' coil latches open (FALSE) when 'powered up', and does not set (to TRUE) when power is removed.

This arrangement is an example of a '**wired OR**'. **Rung 5** shows an alternative approach.

11.2.6 Rung 5 - example

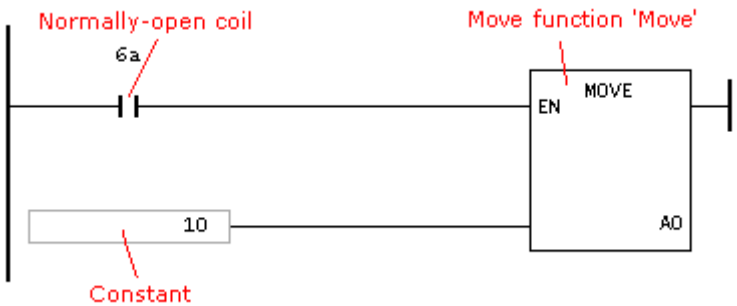


Rung 5 performs the following operation:

If digital 5a is TRUE, OR digital 5b is FALSE, then execution jumps to '_End' (i.e. after Rung 7), otherwise, execution continues normally at Rung 6.

This arrangement uses an explicit **OR function**. **Rung 4** shows an alternative approach.

11.2.7 Rung 6 - example



Rung 6 performs the following operation:

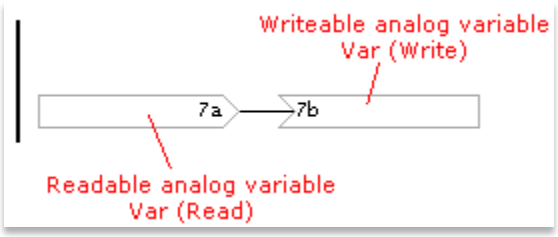
If input 6a is TRUE, the field A0 is updated with the value 10; input 6a is FALSE; the field A0 is not changed.

This arrangement can also be written in Structured Text, as an If-statement;

```
If 6a Then
A0:=10;
```

```
End_If;
```

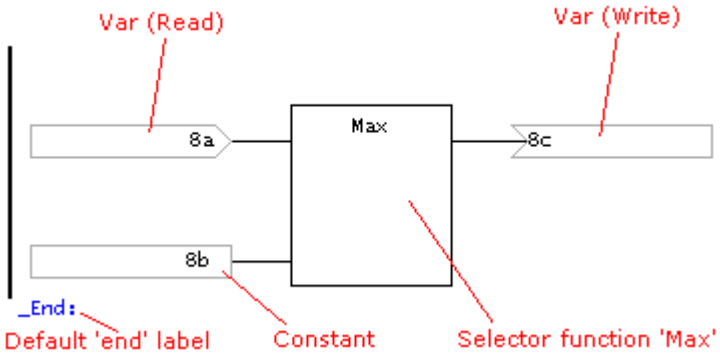
11.2.8 Rung 7 - example



Rung 7 performs the following operation:

Analogue variable 7a is written to analogue variable 7b.

11.2.9 Rung 8 - example

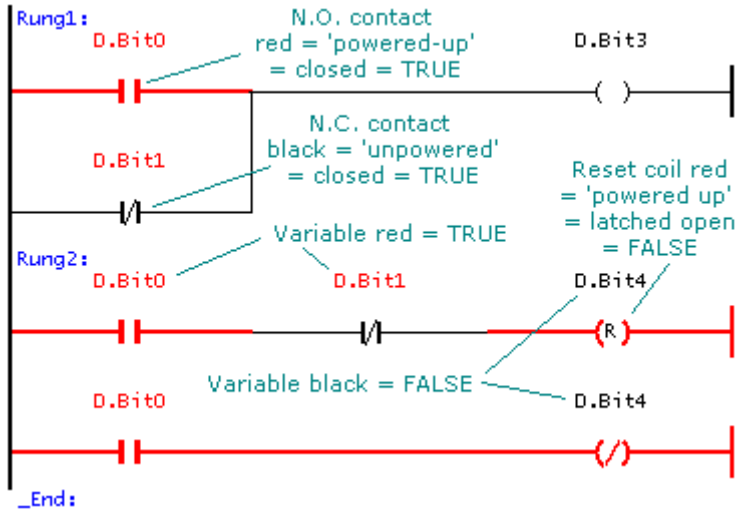


Rung 8 performs the following operation:

The maximum of analogue variable 8a and constant 8b is written to analogue variable 8c.

11.2.10 Ladder Diagram default colours - example

This is an example showing the default colours used in Ladder Diagrams.



12 STRUCTURED TEXT

Structured Text (ST) is the language used to write the statements and expressions that specify text Actions and Transitions in the Sequence Editor or 'non-sequence' LIN Actions that can be created in the Action Editor, e.g. for use with an ACTION block.

ST is composed of written statements separated by semicolons. The statements use predefined syntax configuration and program subroutines to change variables. The variables can be explicitly defined values, internally stored variables, or inputs and outputs. Spaces can be used to separate statements and variables, although they are not necessary.

Note: Structured Text is not case sensitive, but it can be useful to make variables lower case, and make statements upper case. Indenting and Comments should also be used to clarify each Statement in the program.

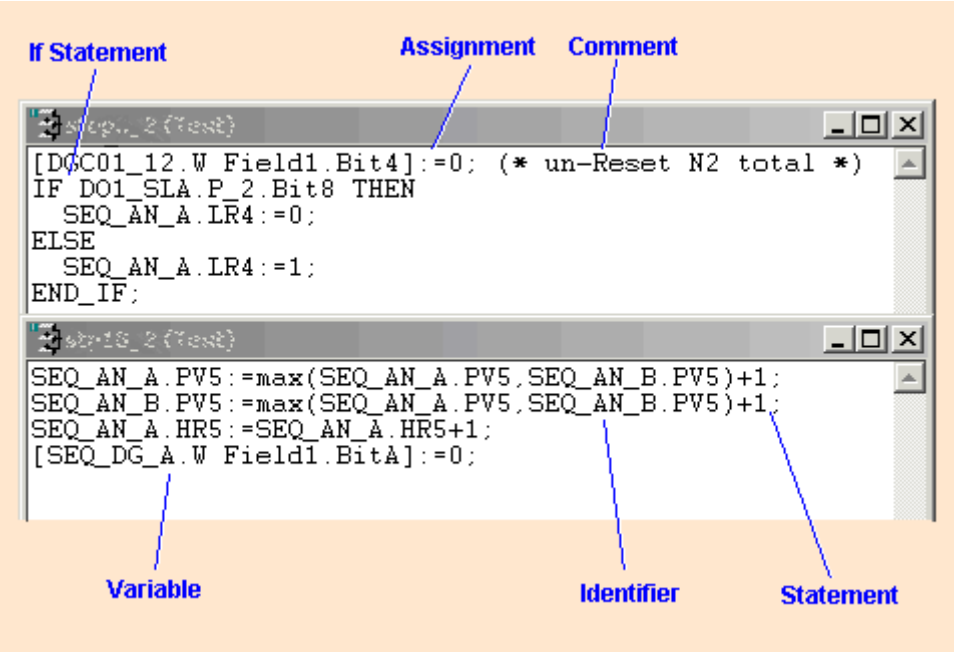
Example:

```
(*ADJUST TEMP SETPOINT*) TIC_100.SL:= REC1.A4;(*A statement*)
```

```
Dig_A.Out AND 64 (*An expression*)
```

12.1 STRUCTURED TEXT – EXAMPLE

This is an example of Structured Text used for configuring an action in a LIN Sequence.



12.2 SPACES IN ST

12.2.1 Mandatory spaces

Space characters must separate logical operators from their operands.

`LEV.OP>15 AND TEMP.OP<55.5` is correct.

12.2.2 Illegal spaces

Spaces must not be left anywhere inside operators, database names, identifiers, or constants.

`HEAT_PID. SL:=REC1. A3;` is incorrect.

12.2.3 Optional spaces

Spaces other than mandatory or illegal spaces may be inserted at will to improve the clarity of the expression or statement.

`TIC_100.PV := 45.5;` needs no spaces but may be clearer with the two shown.

Note: Space characters occupy the same amount of memory as other characters.

12.3 COMMENTS IN ST

(* character string ***)**

Non-executing character strings can be added as 'comments' in the Structured Text of an Action or Transition, e.g. to clarify its purpose. Comments can occupy separate lines, or precede or follow a statement/expression on the same line, or be inserted anywhere that an optional space can go.

```
(*ADJUST TEMP SETPOINT*) TIC_100.SL:= REC1.A4;(*RECIPE #5*)
```

12.4 STATEMENTS IN ST

A statement can be an Assignment, an IF-statement, a FOR-statement, a WHILE-statement, or a REPEAT-statement.

Note: Statements must be terminated by a semicolon.

12.4.1 Assignment

```
variable := expression;
```

Various types of assignment are possible, including the following examples:

ASSIGN A CONSTANT TO A REAL VARIABLE.

```
PIC_004.PV:=35.5;
```

ASSIGN A CALCULATION TO A REAL VARIABLE.

```
ANOP_3.PV:=Const.A6 + (Block_5.PV+3.5) / Const.A7;
```

ASSIGN A CONSTANT TO A DIGITAL VARIABLE.

```
Dig_B.Out.Bit0:=1;
```

ASSIGN A CALCULATION TO A DIGITAL VARIABLE.

```
SEQ4.Hold:=(PIC_005.PV > 30) AND Dig_B.Out.Bit5;
```

ASSIGN A STRING TO A MODE.

```
PIC_004.Mode:="CASCADE";
```

ASSIGN AN INTEGER TO A BITFIELD.

```
Dig_A.Out:=96;
```

ASSIGN A BITWISE CALCULATION TO A BITFIELD.

```
  Dig_A.Out:=Dig_B.Out XOR 96;
```

12.4.2 IF Statement

If Statements are entered as part of a Statement. They include;

IF *expression* **THEN** *statement-list*

{ELSIF *expression* **THEN** *statement-list***}**…
{ELSE *statement-list***}**
END_IF;

Note: The IF-statement allows zero or more ELSIF clauses, and zero or one ELSE clause. IF-statements can be nested, but MUST be closed using END_IF;.

EXAMPLE:

```
IF PIC_004.PV < 10 THEN
     Dig_B.Out.Bit0 := 1;
     RECORD.A4 := ANIN_004.HR;
ELSE
     Dig_B.Out.Bit1 := 1;
END_IF;
```

12.4.3 FOR Statement

Each FOR Statement can be used to perform a calculation on an array of values read from a defined variable.

FOR Statements are entered as part of a Statement.

FOR *variable:=expression* **TO** *expression* **[BY** *literal***] DO**

statement

END_FOR;

EXAMPLE

The following example shows the FOR-Statement used to calculate the average of values read from an array of values. It loops five times adding the array of values, dividing the sum to get the average.

```
avg:=0;
FOR PIC_004.PV := 0 TO 10 BY 2 DO
     avg := avg+f[PIC_004.PV]; (* offset value *)
END_FOR;
  avg:=avg/5; (* Sum divided to calculate the average *)
```

Note: The FOR-statement MUST be closed using END_FOR;.

See also 'Exit Statement'

12.4.4 REPEAT Statement

Each REPEAT Statement can be used to repeat a configured operation continuously until a defined value is obtained.

REPEAT Statements are entered as part of a Statement:

REPEAT

statement-list

UNTIL *expression*;

END_REPEAT;

The following example shows the REPEAT-Statement used to increase the variable value until it reaches 10 or more, defined by the UNTIL statement.

```
REPEAT
     PIC_004.PV:= PIC_004.PV+1; (*array value *)
     UNTIL PIC_004.PV >= 10; (* repeat until variable reaches 10 or more
     *)
END_REPEAT;
```

Note: The REPEAT-statement MUST be closed using END_REPEAT;

See also 'Exit Statement'

12.4.5 WHILE Statement

Each WHILE Statement can be used to perform a calculation on an array of manually assigned values read from a defined variable.

WHILE Statements are entered as part of a Statement.

It includes the;

WHILE *expression* **DO**

statement

END_WHILE;

The following example shows the WHILE-Statement used to calculate the average of five values in a real array. It will loop five times adding the array values, dividing the sum to get the average.

Example:

```
avg:=0;
     PIC_004.PV := 0;
WHILE PIC_004.PV<5 DO
     avg := avg+f[PIC_004.PV]; (* array/offset value *)
     PIC_004.PV := PIC_004.PV+1;
END_WHILE;
avg:=avg/5; (* Sum divided to calculate the average *)
```

Note: The WHILE-statement MUST be closed using END_WHILE;

See also 'Exit Statement'

12.4.6 Exit Statement

An exit statement is used in ‘For’, ‘Whle’ and ‘Repeat’ loops to terminate iterations before the termination condition is satisfied. For example:

```
WHILE <expression 1> DO
   <Statements>
   IF <expression 2> THEN
      EXIT;
   END_IF;
END_WHILE;
```

On Exit from a ‘For’ loop, the value of the loop index is preserved.

When the exit is located within nested iterative loops, control passes to the next statement after the first 'END_FOR', 'END_WHILE' or 'END_REPEAT' loop terminator.

12.5 EXPRESSIONS IN ST

Expressions can contain operators, functions, variables, and constants. Brackets are allowed and can be nested to any level.

The evaluation order of an expression is shown in the Operators & Functions table.

ARITHMETICAL

```
(Block_5.PV + 3.5) / Const.A7 + Const.A6
COS(Block_6.PV)**1.5
Dig_A.Out AND 64
```

LOGICAL

```
Recipe.A0 < 4
Dig_B.Out.Bit5 = Dig_A.Out.Bit3
STEP3.T >= T#4m
A0{7}
```

Note: `A0{7}` shows an example of an offset expression. This offsets a value in the defined field by the number specified in '{}', e.g. `A0{7}` refers to A7, and if N is 6, `[Word 3.Bit4]{N}` refers to `[Word 3.Bit10]` and `[Word 1]{N/2}.Bit4{N}` refers to `[Word 4.Bit10]`.

12.6 OPERATORS AND FUNCTIONS IN ST

In the table below, A, B, C,… etc., are appropriate operands or arguments. Use the correct number format in Structured Text (ST) expressions, and the correct units in trigonometric functions.
The table lists the Operators and also refers to a separate Function Table shown after the 'NOT' Operator. This is for purpose of clarity to distinguish between Operators and Functions and also to conform to the statement in the following Note.

Note: In an expression, the evaluation order of operators and functions follows the table, with the earliest-evaluated operators at the top.

Operator	Purpose	Format
( )	Brackets (alter precedence)	(A+B)*(C+D)
- (negate)	Negate	-A
NOT	Logical NOT (invert of boolean expression)	NOT(A)
FUNCTION	**See Function Table Below**	**See Function Table Below**
******	Exponentiate (Power AB)	A**B[1]
*****	Multiply (×)	A*B
/	Divide (÷)	A/B
MOD	Modulo (remainder of A/B)	(A)MOD(B)[2]
+	Add	A+B
-	Subtract	A-B
=	Equals	A=B
<	Less Than	A<B
>	Greater Than	A>B
<=	Less Than Or Equal To	A<=B
>=	Greater Than Or Equal To	A>=B
<>	Not Equals	A<>B
AND	AND	A AND B [3]
XOR	XOR (exclusive OR)	A XOR B [3]
OR	OR	A OR B [3]
{offset}	Array offset	A{B}
[1] 'A' must exceed zero. [2] 'A' and 'B' must be integers. [3] 16 bit operation		

Note: Some operators are effective only with the appropriate operand types, e.g. the power operator (**) requires A to be positive; the MOD operator must have positive integer operands, etc.

If any STRING is assigned to a Block.Field of either a ByteSeq(BSEQ) or WideString(WSTR) then a $-Conversion will always occur as defined in the rules for Constants in ST.

The Function Table below lists the Sub Categories and Functions in the order listed in the appropriate LINtools ST editor:

Function	Purpose	Format
Sub Category: Convert		
ABS	Absolute	ABS(A)
FLOAT	Convert to Floating Point	FLOAT(A)
ROUND	Round (up or down)	ROUND(A)
TRUNC	Truncate (to an integer)	TRUNC(A)
FORMAT	Formats a value and returns the corresponding byte sequence using the format string to define the conversion required. A null is returned (zero length) if it is not possible to convert the value using the specified format. E.g. Tx_Value = FORMAT("04X", Raw1.I2)converts the integer value in the Raw Comms block register Raw1.I2 to a 4 digit hexadecimal string and writes it to the Tx_Value buffer	ByteSeq FORMAT(ByteSeq, Var Value) Value - The value to be converted Conversion Formats and examples
SCAN	Scans a byte sequence and returns a value using ByteSeq1 to define the conversion required. Leading spaces are ignored (except for the 'C' format) but invalid characters at the start of the byte sequence will cause a zero value to be returned. Invalid characters occurring later in the sequence will cause the conversion to terminate E.g. Setpoint := SCAN("X", MID(Raw1.Rx_Value, 4, 10) would extract the value 453 from the 4 character HEX ASCII value in the byte sequence '[M01G0DI*01C5E5$R' held in the Rx_Value buffer in block Raw1.	VAR SCAN(ByteSeq1, ByteSeq2) ByteSeq1 - A string specifying the conversion format ByteSeq2 - The byte sequence to be converted. Conversion Formats and examples
IS_VALID	Operates in the same way as SCAN but instead of returning a value, returns TRUE if the scanned sequence can be interpreted as a valid number, otherwise FALSE. E.g. IS_VALID("U", ' -4789') would return FALSE since -4789 cannot be interpreted as an unsigned number.	BOOL IS_VALID(ByteSeq1, ByteSeq2) ByteSeq1 - A string specifying the conversion format ByteSeq2 - The byte sequence to be converted. Conversion Formats and examples
Sub Category: Numeric		
ACOS	Arc Cosine (Radians)	ACOS(A)
ASIN	Arc Sine (Radians)	ASIN(A)
ATAN2	Arc Tangent (Radians)	ATAN2(A,B) [Tangent=A/B]
COS	Cosine (Radians)	COS(A)
DEG	Degrees (from Radians)	DEG(A)
EXP	Exponentiation (eA)	EXP(A)
LN	Natural Logarithm (base e)	LN(A)
LOG	Logarithm (base 10)	LOG(A)
RAD	Radians (from degrees)	RAD(A)

SIN	Sine (Radians)	SIN(A)
SQRT	**Square Root**	**SQRT(A)**
TAN	Tangent (Radians)	TAN(A)
Sub Category: Random		
RANDOM	Random value. Not to be used in redundant systems.	RANDOM(A) [A=max modulus]
Sub Category: Selector		
AVG	Average (arithmetic mean)	AVG(A,B,C,...)
MAX	Maximum value	MAX(A,B,C,...)
MIN	Minimum value	MIN(A,B,C,...)
SWITCH	Selects A (integer C=0) else B	SWITCH(A,B,C)
Sub Category: Move		
MOVE (Structured Text(ST))	Assignment	:=
MOVE (Ladder Diagram)	Conditional Assignment	Takes one Boolean and one number, and outputs the number to a specific Variable if the first input (Boolean) is True. If Boolean is False; the number is not moved.
Sub Category: String Basics		
EQUAL	TRUE if the two arguments are identical EQUAL functionality is also available using the "=" operator as above. When '=' is applied to 'STRING' items only, a case-insensitive comparison is applied. The EQUAL function determines strict equality.	BOOL EQUAL(ByteSeq1, ByteSeq2) OR BOOL EQUAL(WideStr1, WideStr2)
LEN	Returns the length in bytes of the byte sequence. If ByteSeq is a null sequence (zero length), a value of 0 is returned.	Int LEN(ByteSeq)
LEFT	Returns L bytes starting from the left end (i.e. from the beginning of the byte sequence). If ByteSeq contains less than L bytes, a null sequence (zero length) is returned.	ByteSeq LEFT(ByteSeq, int L) L - The number of bytes to return.
RIGHT	Returns L bytes from the right end (i.e. up to the end of the byte sequence). If ByteSeq contains less than L bytes, a null sequence (zero length) is returned.	ByteSeq RIGHT(ByteSeq, Int L) L - The number of bytes to return.
MID	Returns L bytes starting from position P. If ByteSeq contains less than L+P-1 bytes, or if P is 0 or negative or greater than the length of ByteSeq, a null sequence (zero length) is returned.	ByteSeq MID(ByteSeq, int L, int P) L - The number of bytes to return. P - The start position from which the bytes to be taken begins.
CONCAT	Returns the concatenation of the two arguments If either argument is of type WSTR then the other is converted to	ByteSeq CONCAT(ByteSeq1, ByteSeq2)

	WSTR.	OR WSTR CONCAT(WSTR1, WSTR2)
INSERT	Returns a sequence of bytes consisting of ByteSeq1 with ByteSeq2 inserted starting at position P. If P is zero, negative, or greater than the length of ByteSeq1+1, a null sequence (zero length) is returned.	ByteSeq INSERT(ByteSeq1, ByteSeq2, int P) P - The position from which to start inserting bytes.
DELETE	Returns a sequence consisting of ByteSeq with L bytes removed starting at position P. If ByteSeq contains less than L+P-1 bytes or P is zero or negative, a null sequence (zero length) is returned.	ByteSeq DELETE(ByteSeq, int L, int P) L - The number of bytes to delete P - The start position from which the bytes to be deleted begins
REPLACE	Returns a sequence of bytes consisting of ByteSeq1 with L bytes replaced by ByteSeq2 starting at position P. If ByteSeq1 contains less than L+P-1 bytes or P is zero or negative, a null sequence (zero length) is returned.	ByteSeq REPLACE(ByteSeq1, ByteSeq2, int L, int P) L - The number of bytes to replace P - The position from which to start replacing bytes
FIND	Returns the position of the beginning of the Nth occurrence of ByteSeq2 inside ByteSeq1. It is permitted to omit N, in which case it defaults to 1. The search is started from the beginning of ByteSeq1 if N is positive, or from the end if N is negative. If N is zero or ByteSeq1 is null or no occurrence is found, 0 is returned. If ByteSeq2 is null, the function returns 1 if N is positive, or the length plus 1 if N is negative. FIND Examples:	Int FIND(ByteSeq1, ByteSeq2, int N) N - The occurrence to find (defaults to 1 if omitted)
DELIMIT	Returns the Nth substring of ByteSeq1 delimited by ByteSeq2. (The start and end of ByteSeq1 also count as delimiters.) N may be negative, in which case it is counted from the right. If ByteSeq1 is null or ByteSeq2 is null or N is 0, or there are less than ABS(N)-1 delimiters in ByteSeq1, a null byte sequence is returned. DELIMIT Examples:	ByteSeq DELIMIT(ByteSeq1, ByteSeq2, int N) N - The occurrence to find
JUSTIFY_LEFT	Justifies a byte sequence within a specified field at the left most position. The return value is padded with spaces on the right to fill the field width. If the length of the argument is greater than FieldWidth, a null sequence (zero length) is returned.	ByteSeq JUSTIFY_LEFT(ByteSeq, int FieldWidth)
JUSTIFY_RIGHT	Justifies a byte sequence within a specified field at the right most position. The return value is padded with spaces on the left to fill the field width. If the length of the argument is greater than FieldWidth, a null sequence (zero length) is returned.	ByteSeq JUSTIFY_RIGHT(ByteSeq, int FieldWidth)
JUSTIFY_CENTRE	Centrally justifies a byte sequence within a specified field. The returned value is padded with equal numbers of spaces on the left and right to fill the field width. If the length of the argument is greater than FieldWidth, a null sequence (zero length) is returned. If the byte sequence cannot be centred exactly it will be biased to the left.	ByteSeq JUSTIFY_CENTRE(ByteSeq, int FieldWidth)
LTRIM	Returns a copy of ByteSeq with any leading spaces removed.	ByteSeq LTRIM(ByteSeq)
RTRIM	Returns a copy of ByteSeq with any trailing spaces removed.	ByteSeq RTRIM(ByteSeq)
TRIM	Returns a copy of ByteSeq with any leading or trailing spaces removed.	ByteSeq TRIM(ByteSeq)
UCASE	Returns a copy of ByteSeq with any lower case ASCII letters converted to upper case.	ByteSeq UCASE(ByteSeq)
LCASE	Returns a copy of ByteSeq with any upper case ASCII letters converted to lower case.	ByteSeq LCASE(ByteSeq)

STRING	Returns a byte sequence consisting of the first character of ByteSeq repeated L times. If L is greater than 1020 it is clipped to 1020.	ByteSeq STRING(int L, ByteSeq) L - The length of the returned sequence
SPACE	Returns a byte sequence consisting of L spaces. If L is greater than 1020 it will be clipped to 1020.	ByteSeq SPACE(int L) L - The length of the returned sequence
STRCOMP	Performs a binary comparison on the two arguments and returns -1 if ByteSeq1 is smaller than ByteSeq2, 0 if they are equal and 1 if ByteSeq1 is greater than ByteSeq2. For example, 'aa' < 'ab' and 'aa' < 'aaa' Note: the STRCOMP functionality is also available using the existing "<", "=", ">", etc. operators.	Int STRCOMP(ByteSeq1, ByteSeq2)
PURGE	Returns ByteSeq1 with any bytes that appear in ByteSeq2 removed. Example: PURGE('abcdabcde', 'bd') returns 'acace'	ByteSeq PURGE(ByteSeq1, ByteSeq2)
DATEAND TIME NOW	Returns an ASCII byte-sequence giving the current date and time in the form "dd/mm/yyyy hh:mm:ss" (i.e. 19 characters). Other functions could then be used to change this to other formats; for example "mm/dd/yy" could be obtained as follows: Buffer1 := DATEANDTIMENOW( ); Buffer1 := CONCAT(CONCAT(MID(Buffer1,3,4), LEFT(Buffer1,3)), MID(Buffer1,2,9);	ByteSeq DATEANDTIMENOW(int mode) mode0 = Local Time mode1 = System Time
Sub Category: Extract		Note: If an attempt is made to EXTRACT an element which is beyond the end of the byte sequence in a buffer, a value of zero is returned.
EXTRACT_BOOL	Extracts a boolean value from a ByteSeq. The value returned is the boolean value. Eight Booleans are stored in each byte.	BOOL EXTRACT_BOOL(ByteSeq, Int P, Int B) P - The byte location to read from B - The bit position within the byte (0-7)
EXTRACT_INT8	Extracts a signed 8 bit integer value located in a ByteSeq. The value returned is the integer value. Each 8 bit integer is stored in 1 byte.	Int EXTRACT_INT8(ByteSeq, Int P) P - The byte location to start from
EXTRACT_UINT8	Extracts an unsigned 8 bit integer value located in a ByteSeq. The value returned is the integer value. Each 8 bit integer is stored in 1 byte.	Int EXTRACT_UINT8(ByteSeq, Int P) P - The byte location to start from
EXTRACT_INT16	Extracts a signed 16 bit integer value located in a ByteSeq. The value returned is the 16 bit integer. Each 16 bit integer is stored in 2 bytes.	Int EXTRACT_INT16(ByteSeq, Int P, Int Order) P - The byte location to start from Order - Byte order option
EXTRACT_UINT16	Extracts an unsigned 16 bit integer value located in a ByteSeq. The value returned is a 32-bit integer containing the 16-bit unsigned value (Note: ST evaluation in LIN does not have a 16-bit unsigned type). Each 16 bit integer is stored in 2 bytes of the buffer	Int EXTRACT_UINT16(ByteSeq, Int P, Int Order) P - The byte location to start from Order - Byte order option
EXTRACT_INT32	Extracts a signed 32 bit integer value located in a ByteSeq. The value returned is the 32 bit integer. Each 32 bit integer is stored in 4 bytes. (Note that in practice this can also be used with 32-bit unsigned values.)	Int EXTRACT_INT32(ByteSeq, Int P, Int Order) P - The byte location to start from Order - Byte order option
EXTRACT_REAL	Extracts a 32 bit real value located in a ByteSeq. The value returned is the 32 bit real. Each 32 bit real is stored in 4 bytes.	Real EXTRACT_REAL(ByteSeq, Int P, Int Order) P - The byte location to start from

		Order - Byte order option
Sub Category: Replace		Note: REPLACE cannot be used to initialise a character buffer. If an attempt is made to write an element which is beyond the end of the byte sequence in the buffer, a null sequence will be returned.
REPLACE_BOOL	Replaces a boolean value located in a character buffer. The value returned is the byte sequence containing the replaced value. Eight Booleans are stored in each byte.	ByteSeq REPLACE_BOOL(ByteSeq, Int P, Int B, BOOL Value) P - The byte location to write to B - The bit position within the byte Value - Boolean value to write
REPLACE_INT8	Replaces a signed 8 bit integer value located in a character buffer. The value returned is the byte sequence containing the replaced value. Each 8 bit integer is stored in 1 byte.	ByteSeq REPLACE_INT8(ByteSeq, Int P, Int Value) P - The byte location to start from Value - Integer value to write
REPLACE_UINT8	Replaces an unsigned 8 bit integer value located in a character buffer. The value returned is the byte sequence containing the replaced value. Each 8 bit integer is stored in 1 byte. (Note that in practice this is identical to REPLACE_INT8.)	ByteSeq REPLACE_UINT8(ByteSeq, Int P, Int Value) P - The byte location to start from Value - Integer value to write
REPLACE_INT16	Replaces a signed 16 bit integer value located in a character buffer. The value returned is the byte sequence containing the replaced value. Each 16 bit integer is stored in 2 bytes.	ByteSeq REPLACE_INT16(ByteSeq, Int P, Int Value, Int Order) P - The byte location to start from Value - Integer value to write Order - Byte order option
REPLACE_UINT16	Replaces an unsigned 16 bit integer value located in a character buffer. The value returned is the byte sequence containing the replaced value. Each 16 bit integer is stored in 2 bytes. (Note that in practice this is identical to REPLACE_INT16.)	ByteSeq REPLACE_UINT16(ByteSeq, Int P, Int Value, Int Order) P - The byte location to start from Value - Integer value to write Order - Byte order option
REPLACE_INT32	Replaces a signed 32 bit integer value located in a character buffer. The value returned is the byte sequence containing the replaced value. Each 32 bit integer is stored in 4 bytes. (Note that in practice this can also be used with 32-bit unsigned values.)	ByteSeq REPLACE_INT32(ByteSeq, Int P, Int Value, Int Order) P - The byte location to start from Value - Integer value to write Order - Byte order option
REPLACE_REAL	Replaces a 32 bit real value located in a character buffer. The value returned is the byte sequence containing the replaced value. Each 32 bit real is stored in 4 bytes.	ByteSeq REPLACE_REAL(ByteSeq, Int P, Int Value, Int Order) P - The byte location to start from Value - Integer value to write Order - Byte order option
Sub Category: Bit		

BIT_INVERT	Returns a 32-bit integer value which is the one's complement of Int. (This will effectively be 16-bit if subsequently used in a 16-bit context.)	Int BIT_INVERT(Int)
SHL8	Returns the integer value (0-255) Int left shifted by N bits, zero filled on the right.	Int SHL8(Int, Int N) Int - 8 bit integer value N - The number of bit positions to shift
SHL16	Returns the integer value (0-65535) Int left shifted by N bits, zero filled on the right.	Int SHL16(Int, Int N) Int - 16 bit integer value N - The number of bit positions to shift
SHL32	Returns the integer value (-2,147,483,648 to 2,147,483,647) Int left shifted by N bits, zero filled on the right.	Int SHL32(Int, Int N) Int - 32 bit integer value N - The number of bit positions to shift
SHR8	Returns the integer value (0-255) Int right shifted by N bits, zero filled on the left.	Int SHR8(Int, Int N) Int - 8 bit integer value N - The number of bit positions to shift
SHR16	Returns the integer value (0-65535) Int right shifted by N bits, zero filled on the left.	Int SHR16(Int, Int N) Int - 16 bit integer value N - The number of bit positions to shift
SHR32	Returns the integer value (-2,147,483,648 to 2,147,483,647) Int right shifted by N bits, zero filled on the left.	Int SHR32(Int, Int N) Int - 32 bit integer value N - The number of bit positions to shift
ROL8	Returns the integer value (0-255) Int left rotated by N bits.	Int ROL8(Int, Int N) Int - 8 bit integer value N - The number of bit positions to rotate
ROL16	Returns the integer value (0-65535) Int left rotated by N bits.	Int ROL16(Int, Int N) Int - 16 bit integer value N - The number of bit positions to rotate
ROL32	Returns the integer value (-2,147,483,648 to 2,147,483,647) Int left rotated by N bits.	Int ROL32(Int, Int N) Int - 32 bit integer value N - The number of bit positions to rotate
ROR8	Returns the integer value (0-255) Int right rotated by N bits.	Int ROR8(Int, Int N) Int - 8 bit integer value N - The number of bit positions to rotate
ROR16	Returns the integer value (0-65535) Int right rotated by N bits.	Int ROR16(Int, Int N) Int - 16 bit integer value N - The number of bit positions to rotate
ROR32	Returns the integer value (-2,147,483,648 to 2,147,483,647) Int right rotated by N bits.	Int ROR32(Int, Int N) Int - 32 bit integer value N - The number of bit positions to rotate
Sub Category:		

Checksum		
BCC	Mode=0 - Returns a checksum byte which is either the exclusive OR of all bytes in the sequence. Mode=1 - The 8 LS bits of the sum of all bytes in the sequence. Mode=2 - The sum, not truncated to 8 bits. Note: An initial 8-bit seed value can be specified if required – if not required it should be set to 0.	Int BCC(ByteSeq, Int Mode, Int Seed) Mode - Mode (XOR, Sum or Untruncated Sum) Seed - Initial seed value
CRC	Returns a 2 byte CRC calculated for all the bytes in ByteSeq. Poly specifies the polynomial coding value and is a binary value containing a bit '1' for each non-zero polynomial coefficient. The most significant polynomial bit (x^16) is omitted. Mode=0 - Defaulft (Non reversed). Mode=1 - the bits within each byte are reversed before performing the calculation. This 'bit-reflection' is a common, though not universal, requirement for CRCs used in comms protocols. This is often the order in which the bits are transmitted. An initial 16-bit seed value can be specified if required – if not required it should be set to 0. Example CRC	Int CRC(ByteSeq, Int Mode, Int Poly, Int Seed) Mode - Mode (whether bit-reflected) Poly - Value of the CRC polynomial Seed - Initial seed value

FUNCTION Table

12.6.1 Example CRC

Returns a 2 byte CRC calculated for all the bytes in ByteSeq.

Poly specifies the polynomial coding value and is a binary value containing a bit '1' for each non-zero polynomial coefficient. The most significant polynomial bit (x^16) is omitted.

Mode=0 - Non-reversed

Mode=1 - the bits within each byte are reversed before performing the calculation. This 'bit-reflection' is a common, though not universal, requirement for CRCs used in comms protocol. This is often the order in which the bits are transmitted.

An initial 16-bit seed value can be specified if required – if not required it should be set to 0.

EXAMPLES OF POLYNOMIALS

CRC16

x^16 + x^15 + x^2 + 1 1-1000-0000-0000-0101

Poly = 8005 hexadecimal

CRC16 reversed

x^16 + x^14 + x + 1 1-0100-0000-0000-0011

Poly = 4003 hexadecimal

EXAMPLE CRC (MODBUS)

CRC('123456789',1,16#8005,16#FFFF) returns 16#4B37

12.6.2 FIND Examples

Returns the position of the beginning of the Nth occurrence of ByteSeq2 inside ByteSeq1. It is permitted to omit N, in which case it defaults to 1. The search is started from the beginning of ByteSeq1 if N is positive, or from the end if N is negative. If N is zero or ByteSeq1 is null or no occurrence is found, 0 is returned. If ByteSeq2 is null, the function returns 1 if N is positive, or the length plus 1 if N is negative, e.g.:

FIND('ABACA','A',2) returns 3

FIND('ABACABA','AB',2) returns 5

FIND('AAAAA', 'AA', -1) returns 4

FIND('AAAAA', 'A', -1) returns 5

FIND('AAAAA','',-1) returns 6

12.6.3 Conversion Formats and examples

Format strings consist of an optional width followed conversion character(s) as defined below. An optional further character in the case of FORMAT will override the default '*' used for unformattable values, e.g. numbers which will not fit within the specified width.

For FORMAT with a specified width, the output is padded on the left with spaces to fill this width, unless the width begins with '0', in which case '0' is used. For SCAN or IS_VALID with a specified width, the input is effectively truncated to this width before being scanned. So, for example, SCAN("6I", Buffer) is equivalent to SCAN("I", LEFT(Buffer, 6)).

For FORMAT of real values only, the width may include an optional decimal point followed by an optional precision (i.e. number of decimal places).

CONVERSION SPECIFICATIONS

I (SIGNED DECIMAL INTEGER).

Examples:

FORMAT("5I",Value) would return the byte sequence ' 7412' if Value was 7412.

FORMAT("05I",Value) would return the byte sequence '07412' if Value was 7412.

FORMAT("05I",Value) would return the byte sequence '-0123' if Value was -123 (FF85 hexadecimal)

FORMAT("3I!",Value) would return the byte sequence '!!!' if Value was 7412.

SCAN("I", ' 4789') would return the value 4789

SCAN("I", ' -4789') would return the value -4789

SCAN("3I", '-4789') would return the value -47

IS_VALID("I", ' -4789') would return TRUE

IS_VALID("I", ' A123') would return FALSE

U (UNSIGNED DECIMAL INTEGER).

Examples

FORMAT("5U",Value) would return the byte sequence ' 7412' if Value was 7412.

FORMAT("05U",Value) would return the byte sequence '07412' if Value was 7412

FORMAT("5U",Value) would return the byte sequence '65535' if Value was FFFF hexadecimal.

SCAN(“U”, ‘4789’) would return the value 4789

SCAN(“2U”, ‘4789’) would return the value 47

SCAN(“U”, ‘-4789’) would return the value 0 as -4789 cannot be interpreted as an unsigned number.

IS_VALID(“U”, ‘-4789’) would return FALSE

x, X (HEXADECIMAL INTEGER).

x for 0 – 9 and a,b,c,d,e,f

X for 0 – 9 and A,B,C,D,E,F

Examples

FORMAT(“06x”,Value) would return the byte sequence ‘007fff’ if Value was 32767 decimal.

FORMAT(“X”,Value) would return the byte sequence ‘FFFF’ if Value was 65535 decimal.

SCAN(“X”, ‘FF’) would return the value 255 decimal

B (BINARY INTEGER).

Examples

FORMAT(“08B”,Value) would return the byte sequence ‘01011101’ if Value was 93 decimal.

SCAN(“B”, ‘ 1111111111’) would return the value 1023 decimal

O (OCTAL INTEGER).

Examples

FORMAT(“O”,Value) would return the byte sequence ‘777’ if Value was 511 decimal.

FORMAT(“4O”,Value) would return the byte sequence ‘ 777’ if Value was 511 decimal.

FORMAT(“04O”,Value) would return the byte sequence ‘0777’ if Value was 511 decimal.

SCAN(“O”, ‘77’) would return the value 63 decimal

F REAL (FLOATING POINT).

For FORMAT only, the size can be followed by an optional decimal point and an optional precision (i.e. the number of digits after the decimal point). Exponential formats are not supported for ‘F’, but see Conversions ‘e, E’ & ‘G’ below.

Examples

Where Value was the result of dividing 1000 by 3

FORMAT(“6.2F”,Value) would return the byte sequence ‘333.33’

FORMAT(“6.F”,Value) would return the byte sequence ‘ 333.’

FORMAT(“6F”,Value) would return the byte sequence ‘ 333’

FORMAT(“F”,Value) would return the byte sequence ‘333’

SCAN(“F”, ‘ -1.456E-2’) would return the value -1.456

e, E

For use only with FORMAT, this behaves like ‘F’ but outputs in scientific format using ‘e’ or ‘E’ as specified. The exponent will include a sign and at least two digits.

Example

Where Value was the result of dividing 1000 by 3

FORMAT(“10.2E”,Value) will return the byte sequence ‘ 3.33E+02’

Note: if 'e' was used in the example, the returned byte sequence would state ‘ 3.33e+02’

G

For use only with SCAN & IS_VALID, this behaves like ‘F’, except that unlike ‘F’ it will accept scientific format.

Example

SCAN(“G”, ‘ -1.456E-2’) would return the value -0.01456

C (SINGLE ASCII CHARACTER).

For FORMAT this converts a value in the range 0 – 255 into a single ASCII character. For SCAN a single character is converted into a value in the range 0 – 255.

Examples

FORMAT(“C”,Value) would return the single byte ‘A’ if Value was 65 decimal.

SCAN(“C”, ‘B’) would return the value 66 decimal.

T ISO8601 FORMAT

Converts between time held as an integer in the ISO8601 format and HH:MM:SS. In ISO8601 time is held as a decimal number in the format HHMMSS.

Can be followed by an optional separator character which is assumed to be ‘:’ if not specified.

Examples

FORMAT(“T”,Value) would return the character sequence ’12:33:06’ if Value was 123306 decimal.

FORMAT(“T.”,Value) would return the character sequence ’12.33.06’ if Value was 123306 decimal.

SCAN(“T”, ’14:27:30’) would return the value 142730 decimal.

TP (Posix format)

Converts between time held as an integer number of seconds and HH:MM:SS format. Time values greater than 86399 will be reduced modulo 86400 so this is compatible with Date/Time values held in Posix format (the number of seconds since midnight on 1/1/70).

Can be followed by an optional separator character which is assumed to be ':' if not specified.

Examples

FORMAT("TP",Value) would return the character sequence '12:33:06' if Value was 45186 decimal.

FORMAT("TP.",Value) would return the character sequence '12.33.06' if Value was 45186 decimal.

SCAN("TP", '00:01:30') would return the value 90 decimal.

D ISO8601 FORMAT

Converts between a date held as an integer in the ISO8601 format and DD/MM/YYYY. In ISO8601 dates are held as decimal numbers in the format YYYYMMDD

Examples

FORMAT("D",Value) would return the character sequence '23/09/2006' if Value was 20060923 decimal.

SCAN("D", '04/07/2005') would return the value 20050704 decimal.

DP Posix format

Converts between time held as an integer number of seconds since midnight 1/1/1970 (Posix format) and date in DD/MM/YY format.

Examples

FORMAT("DP",Value) would return the character sequence '05/01/70' if Value was 345600 decimal.

SCAN("DP", '15/01/70') would return the value 1209600 decimal.

Note that LIN-encoded dates and times, specifically the LIN DATE and TIME types are handled within ST as strings. Therefore LIN encoding is implicit when reading from or writing to DATE or TIME fields.

DELIMIT

Returns the Nth substring of ByteSeq1 delimited by ByteSeq2. (The start and end of ByteSeq1 also count as delimiters.) N may be negative, in which case it is counted from the right. If ByteSeq1 is null or ByteSeq2 is null or N is 0, or there are less than |N| - 1 delimiters in ByteSeq1, a null byte sequence is returned.

Examples:

DELIMIT('a+b+c', '+', 2) returns 'b'

DELIMIT('abc;;def;ghi', ';', 3) returns 'def'

DELIMIT('a::bb::ccc::dddd', '::', -2) returns 'ccc'

DELIMIT('A/B/C/', '/', -4) returns 'A'

DELIMIT('ABCDEFG', 'BX', 1) returns 'ABCDEFG'

DELIMIT('ABCDEFG', 'BX', 2) returns ''

As with FIND, occurrences are treated as non-overlapping, for example

DELIMIT('ABC>>>DEF>>>G','>>',2) returns '>DEF'

ORDER PARAMETER

Some functions have an 'Order' parameter which provides different options for byte or word order in the data value to be written or read. These options include 'big endian' and 'little endian' byte order for integer values and 'normal' and 'swapped' word order for 32 bit values.

Order Parameter Table:

Order	16 bit values	32 bit values
0	Little Endian n LS byte n+1 MS byte	Little Endian n LS byte n+1 n+2 n+3 MS byte
1	Big Endian n MS byte n+1 LS byte	Big Endian n MS byte n+1 n+2 n+3 LS byte
2	NA	Little Endian, word order swapped n MS word, LS byte n+1 MS word, MS byte n+2 LS word, LS byte n+3 LS word, MS byte
3	NA	Big Endian, word order swapped n LS word, MS byte n+1 LS word, LS byte n+2 MS word, MS byte n+3 MS word, LS byte

12.7 VARIABLES IN ST

The types of Variable that can be used are:

- database names
- bitfields
- aliases
- SFC step variables

Note: Identifiers contained in a Variable must be valid.

12.7.1 Database names

These can take the following formats:

- blockname.fieldname e.g. `PIC_004.PV`
- blockname.bitfield.bit e.g. `Dig_A.Out.Bit3`
- blockname.bitfield e.g. `TIC_100.Options`
- blockname.Alarms.alarmname e.g. `ANIN_005.Alarms.HiLevel`

Notes:

1. LIN Database names are not case-sensitive, i.e. capital or small letters may be freely used or mixed in the names. For example, `Dig_A.OUT.BIT3` is equivalent to `Dig_A.Out.Bit3`.
2. Database names can be indexed to create a form of array. See Arrays in ST for more details

12.7.2 Bitfields

8-bit bitfields may be referred to as decimal integers in the range 0 to 255 (i.e. binary 00000000 to 11111111).

For example: `Dig_A.Out:=96;` assigns the binary value 01100000 to the bitfield.

- 16-bit bitfields may be referred to as decimal integers in the range -32768 to +32767 (i.e. binary 1000 0000 0000 0000 to 0111 1111 1111 1111, in 2's complement format).

12.7.3 Aliases

TagName aliases, created in the Tag table, can be used in Structured Text and are displayed, as with all Variables, via the Template Palette or the Variable field PickList, in the Object Properties pane.

12.7.4 SFC step variables

These can take the following formats:

- stepname.X, e.g. `END_STEP.X`
- stepname.T, e.g. `STEP5.T`

Stepname.X is TRUE when the step called Stepname is active, or when the step timers have been stopped but not initialised (Run and Init = FALSE, in the SFC_CON block). Otherwise Stepname.X is FALSE.

Stepname.T is the elapsed time in the step called Stepname, in seconds. Stepname.T is frozen when Stepname is exited, and zeroed when Stepname starts or when the SFC is reset.

12.8 IDENTIFIERS IN ST

An identifier is a blockname, fieldname, or stepname. A valid identifier must have the format:

letter | _ {letter | digit | _ }...

That is, one letter or underscore, followed by any number (including zero) of letters, digits or underscores, up to a maximum of eight characters.

12.8.1 Valid Structured Text (ST) identifier examples

- A1
- act_4
- FlowMetr
- _Pump
- `X__3Y` (consecutive underscores)
- z567931

12.8.2 Invalid Structured Text (ST) identifier examples

- `1A` (digit first)
- `Pump 3` (space)
- `Flow-Mtr` (minus sign)
- `x*y*z` (multiply sign)
- `6366` (digit first)
- `T1.2` (dot)
- `Pulserate` (too long)

Note: Variable names containing bad Identifiers can be made recognisable as Variables by enclosing the whole Variable name in square brackets.

Examples:

```
[6366.MN]
[DG_CONN3.W Field2.Bit6]
[x*y*z.OP]
```

12.9 CONSTANTS IN ST

12.9.1 Structured Text (ST) supports four types of constant:

- Integer
- Real
- Time
- String

INTEGER CONSTANTS

Integers are considered as 32 bit signed Integers. Values outside -2,147,483,648 to 2,147,483,647 are clipped to these limits.

Decimal constants have the format:

{sign} digit {digit}…

`45`

`-12345`

Hexadecimal constants have the format:

16#hexdigit {hexdigit}…

16#FF (represents 255 decimal)

16#00ff (represents 255 decimal - padding zeroes OK)

16#03e8 (represents 1000 decimal)

Note: Hexadecimal digits are not case-sensitive.

Binary constants have the format:

2# binarydigit { binarydigit }…

2#1100 (represents 12 decimal)

2#0010 (represents 2 decimal)

REAL CONSTANTS

Real constants have the format:

{sign} integer.integer

3.6

-0.0033

TIME CONSTANTS

Time constants have the format:

T# {integer unit} real unit

Units are **d** (days) **h** (hours) **m** (minutes) or **s** (seconds). Time constants evaluate to seconds if assigned or compared to real constants, e.g. In the assignment `Recipe.A3:=T#1h;` **Recipe.A3** takes the value **3600**.

T#3s (3 seconds)
T#4m13.0s (4 minutes, 13 seconds)
T#5.6h (5 hours, 36 minutes)
T#1d12.2h (36 hours, 12 minutes)

Note: Time units may be input as upper or lower case.

STRING CONSTANTS

String constants must be enclosed in single or double quotation marks.

```
"MANUAL"
"P"
```

String constants fields

Certain block types fields, can be written to/from ST only by using strings, e.g.

```
pid1.mode:="Manual";
TimeDate.Date1:="18/07/09";
```

The table below lists these block types and associated example fields::

Block Types	Example Fields
STRING	`SFC_CON.FileName:="Batch1"`
ENUM	`pid1.mode:="Manual"`
DATE	`TimeDate.Date1:="18/07/09"`
TIME	`TimeDate.CurrTime:="14:22:18"`
WSTR	`WIDESTR24S.VAL1:="r$F4le"`
BSEQ	`RAW_COM.Buffer1:="a$62c"`

WIDE STRING AND BYTE SEQUENCE CONVERSIONS

With reference to the $-sequence table given below, when a character or byte string is assigned to a Wide String(WSTR) or Byte Sequence(BSEQ) a $- sequence conversion based on IEC61131-3 occurs as shown in the examples below.

Note: Byte Sequence and Wide String Fields cannot be wired

Examples:

`RAW_COM.Buffer1:="a$62c"` --------- Assigns 'abc' to the RAW_COM block, Buffer1 field. ($62 = 'b' ASCII)

`WIDESTR24S.VAL1:="r$F4le"` --------- Assigns 'rôle' to the WIDESTR block, Var1 field. ($F4 = 'ô' ASCII)

`Block.Field(STRING):="a$62c"` --------- Assigns 'a$62c' to the block, field. (No Conversion occurs for 'STRING' fields)

$-Sequence	Conversion
$nn	The character with ASCII code 'nn' in hexadecimal, e.g. $0A
$$	Dollar sign
$'	Single quote
$L or $l	Linefeed (0A Hexadecimal)
$N or $n	Newline (equivalent to $R followed by $L)
$P or $p	Form feed (0C Hexadecimal)
$R or $r	Carriage Return (0D hexadecimal)
$T or $t	Tab (09 Hexadecimal)
$U	Reserved for future use

$-Sequence Table

12.10 ACTIONS IN ST

Text Actions are Structured Text (ST) statements that assign values to variables, conditionally or unconditionally. A text Action can contain one or many statements.

- **SFC Actions** contain Steps that may refer to text actions, Ladder Diagram Actions, or to other SFC Actions.

12.11 TRANSITIONS IN ST

Transitions are Structured Text (ST) expressions that may be either TRUE (evaluate to logic or integer 1) or FALSE (evaluate to logic or integer 0).

A Transition is used in an Sequence Action to test for a condition in the control Database. A TRUE Transition following an active step (or parallel steps) causes the Sequence to move on to the next step(s). That is, the active step(s) preceding the Transition de-activate(s), and the step(s) following the Transition activate(s).

- Test a real variable against a constant

```
Recipe.a0 < 4
```

- Test a digital for TRUE

```
Dig_B.Out.Bit5
```

- Test a digital for FALSE

```
NOT Dig_B.Out.Bit5
```

- Test a step time

```
STEP6.T >= T#2m
```

- Test a step time in seconds

```
waitstep.t > 95
```

- Test if an SFC Action end-step reached

```
Act4_end.X
```

- Logical combination of tests

```
Dig_B.Out.Bit5 AND (Rec.A0<4 OR Rec.A0>5) OR step6.T>=T#4s
```

12.12 ARRAYS IN ST

LIN does not support arrays in ST configuration, but the syntax described below can be used to perform a function that provides a form of array (Offset) configuration using the { and } brackets. The expression in the { and } brackets evaluates to an offset from the current field, to make a reference to another field.

Note: Square brackets are used to indicate invalid id's, e.g. block names with spaces and other characters.

Offset syntax can be used In both Arithmetical and Logical expressions and to the left of assignments:

```
Block.Field {<expression>}

Block.Field.Subfield {<expression>}

Block.Field {<expression>}.Bit0

Block.Field {<expression>}.Subfield {<expression>}
```

This syntax can also be used to treat sets of numbered fields in a block as an array, e.g. an AN_CONN block has fields PV1 to PV8 that can be treated as an array of 8 fields. Any individual

field in a set of numbered fields can be used as the base of the array, i.e. a block with 8 REAL_F fields can be used as two arrays of 4 reals, e.g. PV1{3} means PV4, and PV5{3} means PV8. This can also apply to subfields.

Note: `Block_5.Val1.Bit7{0}` is not a permitted offset value syntax because Bit 7 is the last bit in an 8-bit subfield so therefore cannot be the start of an array.

Example:

```
FOR TC001:=0 to 6 DO (*Analogue shift register*)
  Block_5.Val1{TC001}:=Block_5.Val1{TC001 + 1};
END_FOR;
Block_5.Val1{7}:=50
```

12.13 NOTATION

The following syntax and format notation is used in the Structured Text (ST) reference:

- **Boldface** - Items in boldface characters must be input as shown.
- *Italics* - Items in italics are to be supplied by the user.
- { } - Items inside braces are optional.
- … - Items followed by an ellipsis may be repeated any number of times.
- | - Items separated by a '|' character are alternatives (XOR'd).
- `Courier` - Items printed in Courier typeface are examples of Structured Text (ST).

All punctuation - including periods, commas, colons, semicolons, underscores, quotation marks, parentheses, and equal signs - must be included where shown.

13 ONLINE CONFIGURATION AND RECONFIGURATION

Online Configuration, using the Connect feature to locate a specific Instrument, and Reconfiguration, using Try, Untry, and Apply features, is the facility to make direct LIN Database changes to a Strategy in a target instrument currently operating in a system.

If differences between the LINtools Strategy and target instrument Strategy exist, the LINtools program compares the instruments latest structural edits to assess the Alignment Options. There are two distinct types of changes that can be made.

- Live changes
- Tentative changes

13.1 LIVE CHANGES

'Live' changes are defined as edits made directly to the running Strategy immediately affecting the target Instrument. For example, changing tuning parameters in a PID block happens immediately. In order to make tentative parameter changes to a running Strategy, the block must be detached from the Strategy. The Unlink command allows the block to remain running in the instrument, but is considered as Offline when edited in LINtools. Any changes do not directly apply to the running block, and can be 'Tried' in complete safety, until the results are as required.

13.2 TENTATIVE CHANGES

'Tentative' changes can be defined as edits that modify the structure, such as adding or deleting function blocks, or, moving a wire from one input to another. These changes can be made as 'Tentative' because there is no immediate instruction to the target instrument; the delete and create is executed as 'Tentative' and then the new result 'Applied' to the target.

Note: 'Tentative' changes apply to the object being deleted or created. An existing wire can be marked 'delete this - tentative' and a 'wire added - tentative' offering an object the following states: tentative new; existing permanent; tentative delete.

Where Tentative changes include new Function Blocks, the new blocks can be configured in their Tentative state, as if 'offline' (LIN Database is not running in the target), the configuration rules that are applied will be the same as in the normal offline **Editor**. When those changes are then applied to the target, the blocks become 'Online' and operate as in Monitor mode (LIN Database is running in the target).

Where Tentative changes include deleting or creating wires, the wires are used as their state suggests, i.e. a deleted wire is ignored, while an added wire passes the output from one block to another. When a wire contains multiple connections that differ, LINtools identifies it using colour and line style.

13.2.1 I/O Configuration

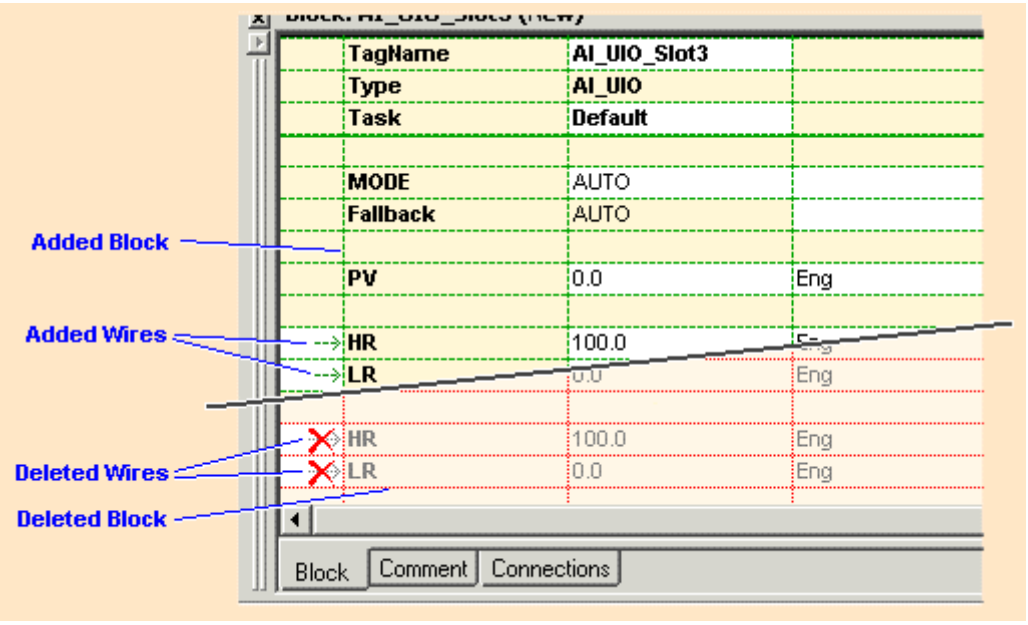
Any structural changes made to the I/O while Online are considered as 'Tentative'. Online reconfiguration of the I/O system involves edits to running blocks (e.g. to add I/O modules or extra channels on the same I/O module), but it is desirable that any changes should be made tentative, i.e. capable of being Tried/Untried, since a change made to one I/O block can potentially affect many others. Changing a module's type, for example, could throw all channel blocks assigned to it into status error (i.e. non-zero value in the Status field). Therefore, when in Online Reconfiguration mode LINtools will not permit modification of certain fields of I/O blocks via the Object Properties pane unless the block is first Unlinked.

Note: If the Strategy configuration files do not already exist on the Computer, press the Connect toolbutton on the toolbar, and select the Port and Node. LINtools can then upload the files from the target Instrument.

13.3 ONLINE RECONFIGURATION – EXAMPLE

This shows the Object Properties pane displaying added and deleted function block diagram (FBD) objects.

Note The Connections tab shows the source and destination of a wire using colour, symbols and line styles to indicate it's state.



13.4 ONLINE RECONFIGURATION WATERMARK

14 PROFIBUS CONFIGURATION

Profibus configuration defines the communication between instruments operating on a Profibus Network, e.g. Eycon™20 (Profibus Master), and other devices (Profibus Slaves).

The Profibus Master configuration runs in conjunction with a LIN database, in an instrument environment provided with a Profibus interface. The interface between the LIN database and the Profibus data uses a GateWay file (.gwf) that configures the LIN instrument to communicate with the Profibus Slaves. This file must be generated and downloaded along with the LIN Database, .dbf, to a LIN instrument.

The relationship between data collected by the Profibus Slave and the associated LIN database fields is configured using the Profibus Master Configurator in LINtools. This software uses the LIN Profibus Master Configuration (.upm) file to link the GWProfM_CON block in the database to the Profibus network. When the configuration is saved a Profibus Binary (.upb) file is generated. This file must also be downloaded to the LIN Profibus Master. The .upb file describes the specific configuration of the network, devices, and informs the Profibus Master what data is expected, based on the parameters defined in the .gsd file, provided by the manufacturer. Each LIN Profibus Master instrument specifies a unique gateway file that is identified via a GWProfM_CON block.

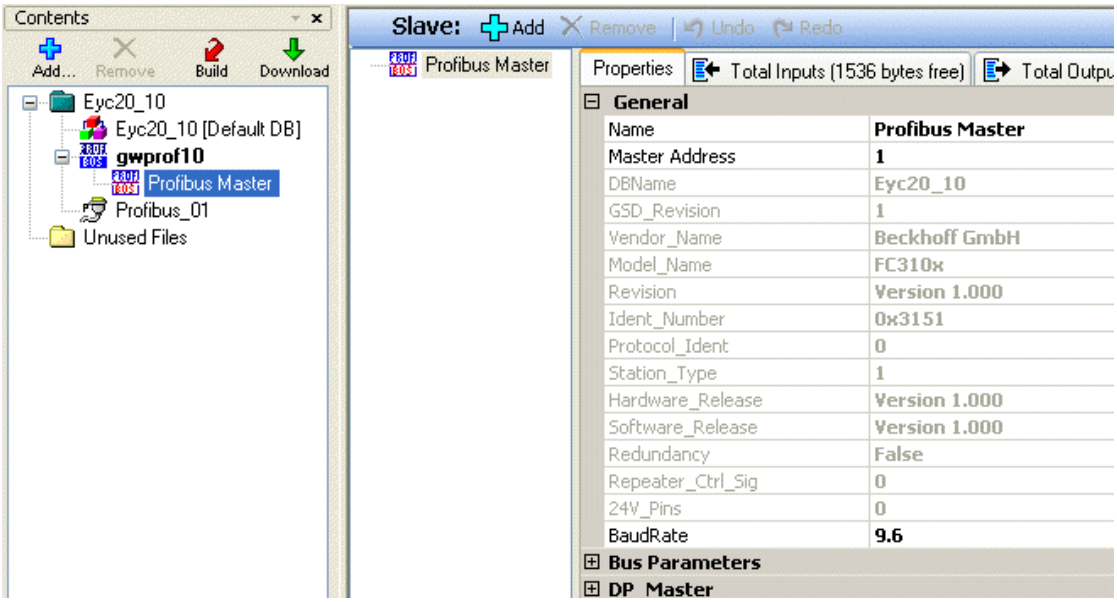
14.1 PROFIBUS MASTER CONFIGURATOR

The primary function of the Profibus Master Configurator is to assign LIN database block fields via the LIN Reference field to a specific address space in the Profibus Slave.

The Profibus Master Configurator displays the Network configuration of Profibus Master, Network and Profibus Slaves, and the configured Modules, in a tree view. Profibus Slaves can be Compact devices or Modular Stations. The Profibus Master Configurator is used to identify each Module fitted to the Profibus Slave. When modules have been added the Profibus Master can read this information and understand what data to expect, e.g. Input data from Analogue Input module and Output data from an Analogue Output module.

The specific Input data and Output data, Extended Diagnostics, and Acyclic data must be configured for each Profibus device via the appropriate pages in the Profibus Master Configurator.

Example:



DEMAND DATA

Demand Data is a sub-protocol using the first eight bytes in both the Input data and Output data of the cyclic Data exchange. It allows random read/write access to any defined parameter in the device.

It is used to define device parameters that are to be transferred at a priority greater than cyclic data without introducing new messages. Support for Demand Data is dependent on the Profibus card, and controlled via the Profibus Master .gsd file.

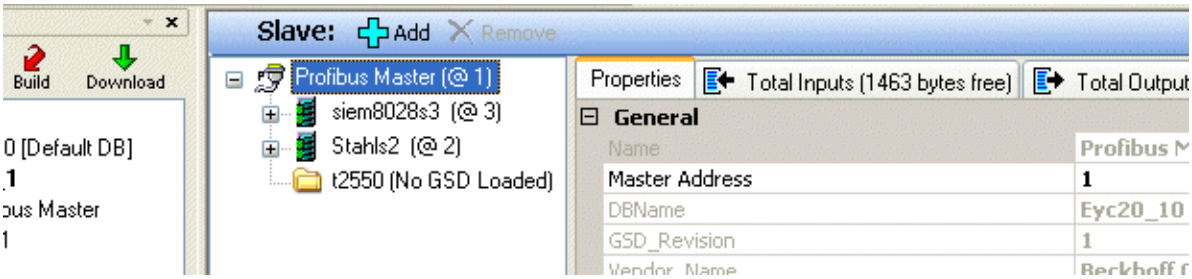
Note: Demand Data will operate correctly only when configured to Module 1.

14.2 PROFIBUS CONFIGURATION – OVERVIEW

Profibus configuration is used for configuring the transfer of data existing in Profibus Slaves, both LIN and third-party devices, to and from a Profibus Master. It allows the user to read and write selected parameters from third-party devices via the Profibus network in to a LIN database.

For Profibus configuration a basic knowledge of the devices that exist on the Profibus network, the configuration of the device, and the parameters that are to be transferred between Profibus Master and Profibus Slave is required. A further understanding of address space for each parameter used is useful.

The Profibus Master Configurator tree view, below, shows the devices connected to this particular Profibus Master. Devices are added to the tree view by using the **Add Slave** button, and removed using the **Remove Slave** button. A further selection of buttons is provided depending on the context of the window beside the tree view. The Input data, Output data, Extended Diagnostics, and Acyclic data are configured on separate pages shown beside the tree view.



To use the Profibus Configurator,

1. Configure the LIN Profibus GateWay.
   This will seamlessly create all required files and folders and launch the Profibus Master Configurator. While configuring the LIN Profibus GateWay, the Network folder can be specified or a new Network folder created.

2. Configure the Profibus communications protocol for the Profibus Master.

   This defines the communications Protocol used by this COM port, i.e. Profibus Master COM port should be configured to ProfibusDvp1-M.

3. Configure the Profibus Master properties.
   These properties define the control of the devices in the network.

4. Configure the Profibus Slave
   Profibus Slaves can be Compact devices or Modular stations and are added to represent each device communicating via the Profibus network connected to the Profibus Master. A .gsd file provides detailed information about the Profibus Slave.
   Configure the Data exchange parameters to define the values that will be read by or written to the Profibus Slave via the Profibus Master.

5. Save and download the Instrument configuration.
   Save the strategy to ensure the information is retained. Add the relevant files to the list of files to be downloaded, and download the strategy to ensure the Instrument operates as expected.

15 DATA RECORDING CONFIGURATION

Data Recording configuration defines which block fields in a database will be recorded. The values from selected block fields are recorded to Data Recording (.uhh) files, and can be displayed at a later date as an electronic chart or spreadsheet display, which can be printed if required.

Note: Recorded data in the .uhh files can be used to replace periods of lost data in InSQL and the Alarm database using the Store and Forward tool.

Data Recording configuration consists of a Data Recording Configuration (.uxg) file, and a Data Recording compound, DataRec. The .uxg file is used to define which block fields are to be recorded, while the DataRec compound controls and monitors the actual recording, e.g. the update rate, and archiving, e.g. PrArchv1.

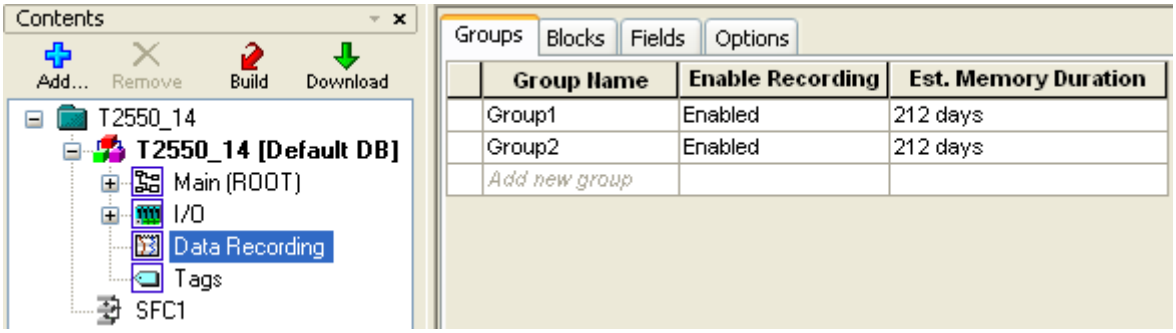
Note: The archiving of recorded .uhh files is configured via the Instrument Properties, and monitored using the RARCDIAG block.

The .uxg file must be downloaded to the instrument.

15.1 DATA RECORDING CONFIGURATOR

The primary function of the Data Recording Configurator is to define groups of block fields in the database, and record the associated values at defined intervals. This provides an accurate record of a particular part of the plant/system.

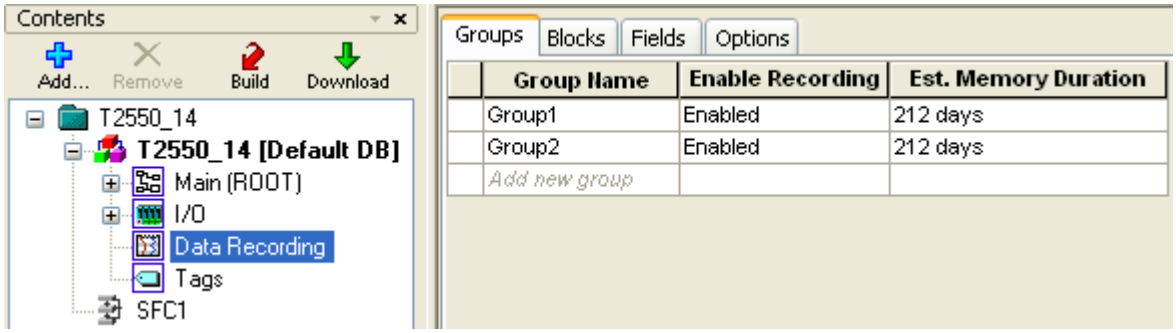
Example:



15.2 DATA RECORDING – OVERVIEW

Data recording is the process of writing groups of values from selected block fields in the database to a Data Recording (.uhh) file. As data is sampled and logged in the instrument, .uhh files are created in the internal flash memory. The .uhh file allows the user to read and print the selected block fields in Review software.

Note: Data Recording requires a D10 to D90 Licence, and supports function block codes K to T. The Licence code is used to permit the use of the blocks, and the function block codes describe a collection of supported function blocks.



Typically, Data Recording is used for

- general audit records (for subsequent analysis with MS Excel, for instance)
- quality control of product and plant
- monitoring performance

The Data Recording configuration functions according to the mode selected in LINtools. Full Data Recording configuration is available when LINtools is operating in Online Reconfiguration, but if LINtools is operating in Connect or Monitor mode only limited functionality is available. Connect mode only allows users to control data recording by enabling or disabling specific groups, and Monitor mode simply displays the Data Recording configuration.

To use the Data Recording Configurator,

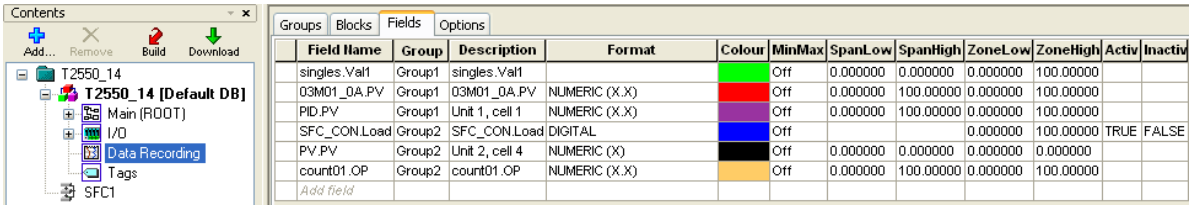
1. Configure the Data Recording configuration.
   Define block fields to be recorded and configure the presentation when displayed as a chart using the Review software.

2. Define the FTP Server.
   Identify the archive Host, FTP Server, used to archive the .uhh files.

Note: Use the *Store and Forward* tool to replace periods of lost data in InSQL and the Alarm databases with recorded data from the .uhh files in LIN instruments and 5000/6000 recorders.

3. Save and download the Instrument configuration.
   Save and download the strategy to the instrument. The Data Recording configuration file, .uxg, will automatically be added to the list of Files to be Downloaded.

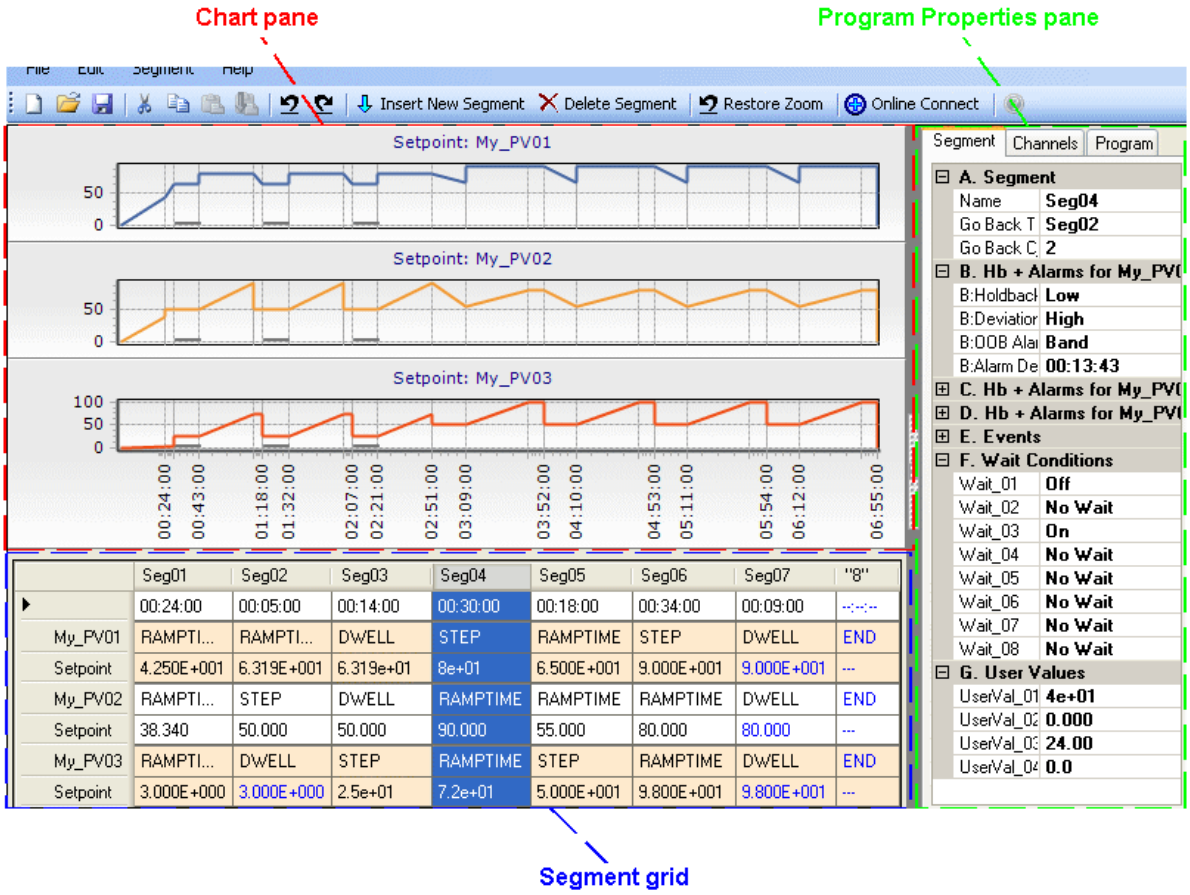
4. Configure the Review software.
   Imports recorded data, .uhh file, into the Review software, and creates charts to display the recorded data.



16 SETPOINT PROGRAMMING

Setpoint Programming is used to control a selected parameter value (channel), typically a Setpoint, over a specified period of time.

The Setpoint Program information is generated by the Programmer Editor and held in a Program file, .uyy. The Program file is produced from a previously configured template defined in a Program Template file, .uyw. This Program Template file is created using the Programmer Wizard available from the Tools menu in LINtools.



The Program Template file defines a suite of blocks that includes the PROGCTRL block, PROGCHAN and SEGMENT blocks, held in a PROG_WIZ compound. These blocks provide access to all parameters in the Setpoint Program that are used to configure the values in the Program file and control the Profile of the setpoint, Events and User Values in the instrument.

Note: The Program configured in the Programmer Editor is read by the blocks in the .dbf file when LINtools is connected to the instrument.

16.1 SETPOINT PROGRAMMING - OVERVIEW

The purpose of a Setpoint Program is to control and manage the changing target value that an automatic control system (e.g. PID controller),aims to reach. For example, a boiler control system might have a temperature setpoint, i.e. a temperature that the control system aims to attain in the system.

The Setpoint Program is a set of defined values for a Single Channel Program or a Multi-Channel Program stored in a Program (.uyy) file. A Program file is generated using this Editor within the specification defined in a Program Template (.uyw) file, generated using the Programmer Wizard.

Note: Programs can be launched using the appropriate command available from the Tools menu.

To create a Program, a Program Template must first be produced using the Programmer Wizard in LINtools. The resulting Program Template file can then be used by the Programmer Editor to create the Setpoint Program. The Programmer Wizard creates blocks that can control only Programs which are produced using the Programmer Editor.

Note: Use the Programmer Wizard in LINtools to generate or edit a .uyw file and create the PROG_WIZ compound. Make sure the required input values and output values to/from the plant/system are wired to the block in this compound.

To configure a Setpoint Program

1. Use the Programmer Wizard to generate or edit a .uyw file, defining the number of channels and segments available to the Program in this instrument, and create all the blocks used to interface with this .dbf file.

2. Open the PROG_WIZ compound, and the PROGCTRL block to show the **Object Properties pane**.

Note: To prevent erroneous Program Template file configuration always use the wizard to edit the blocks in the PROG_WIZ compound. Changing the number of Profiled Channels, Digital Events, or User Values will invalidate any Program file created with the previous version.

i. Wire specific blocks in the PROG_WIZ compound to other blocks in the .dbf file. To use the Program to control the setpoint of a control loop
ii. Wire the control loop configuration (LOOP_PID block) to the Programmer configuration (PROGCHAN block) and return the current setpoint from the Programmer configuration (PROGCHAN block) to the control loop configuration (LOOP_PID block. This will provide the setpoint control for the control loop configuration. Wire the input values (AI_UIO.PV) from the plant/system to the control loop (LOOP_PID.Main.PV).
iii. Wire the Digital Events and User Values defined using the Programmer Wizard to appropriate output blocks.
iv. Wire to the required Wait conditions and Exit conditions defined using the Programmer Wizard from appropriate input blocks. When wiring is complete, save the .dbf file. Add the Program Template file and the Program file to the List of files to be Downloaded.
v. Enter the name for the Program file in the ProgFile field, and click to reveal the context menu. Select the **Open Program file (.UYY)** command. If the file already exists the .uyy file opens in the Programmer Editor, but if it does not the .uyy file is created after pressing Yes to confirm that a new file is required.
vi. Alternatively, launch the Programmer Editor to generate or edit the .uyy file, defining the control of the Setpoint. Save the .uyy file. Select **Tools** > **Programmer Editor**, then open or create a .uyy file.

Note: To configure a .uyy file refer to the Programmer Editor help file.

vii. If necessary, move the required .uyw and .uyy files from the Unused Files folder to the Instrument folder to ensure the files will be downloaded to the instrument.

Note: Use the context menu to display the Unused Files folder if it is not already shown in the Contents pane.

viii. Create and/or open a Program file. This can be done by using the context menu available when selecting the PROGCTRL.File.ProgFile (block.page.field) in the Object Properties pane after providing the Program name, or by opening the Programmer Editor , and selecting **File** > **New (Open)**, and choose the Program Template file that matches the blocks of a PROG_WIZ compound in the database.

3. Configure the Program using the Programmer Editor, setting each Segment type, Duration, and Target Setpoint in the Segment grid as required. Then configure the Digital Event Outputs, User Values, Exit and/or Wait conditions in the Program Properties Pane.

Note: Refer to the Programmer Editor (Tools menu) for full details.

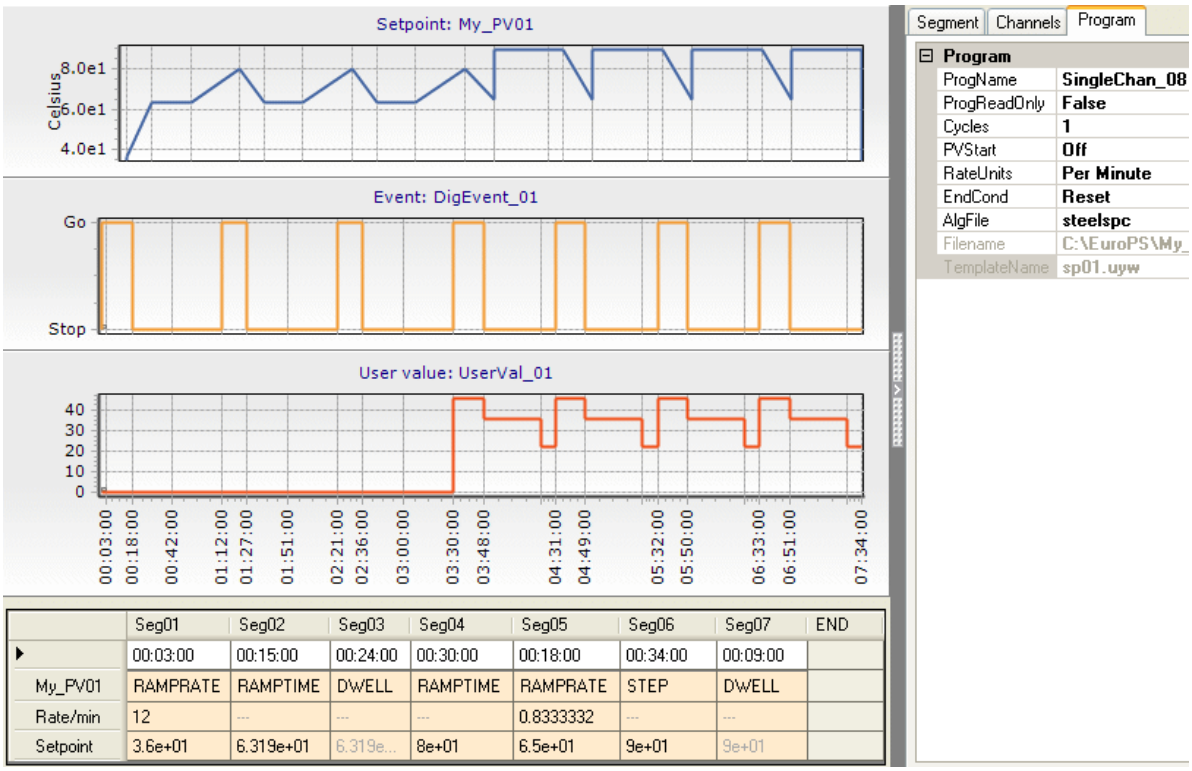
4. Download all relevant files to the instrument using LINtools.

Note: Connect to the instrument from the Programmer Editor to control the running Program.

16.2 SINGLE CHANNEL PROGRAM

A Single Channel Program is suitable for applications where a single Setpoint is profiled, e.g. heat treatment or firing ceramics materials.

The Program type is pre-defined in a Program Template file that is generated by the Programmer Wizard in LINtools. A Single Channel Program creates a single Profile for the Name defined in the Programmer Wizard.



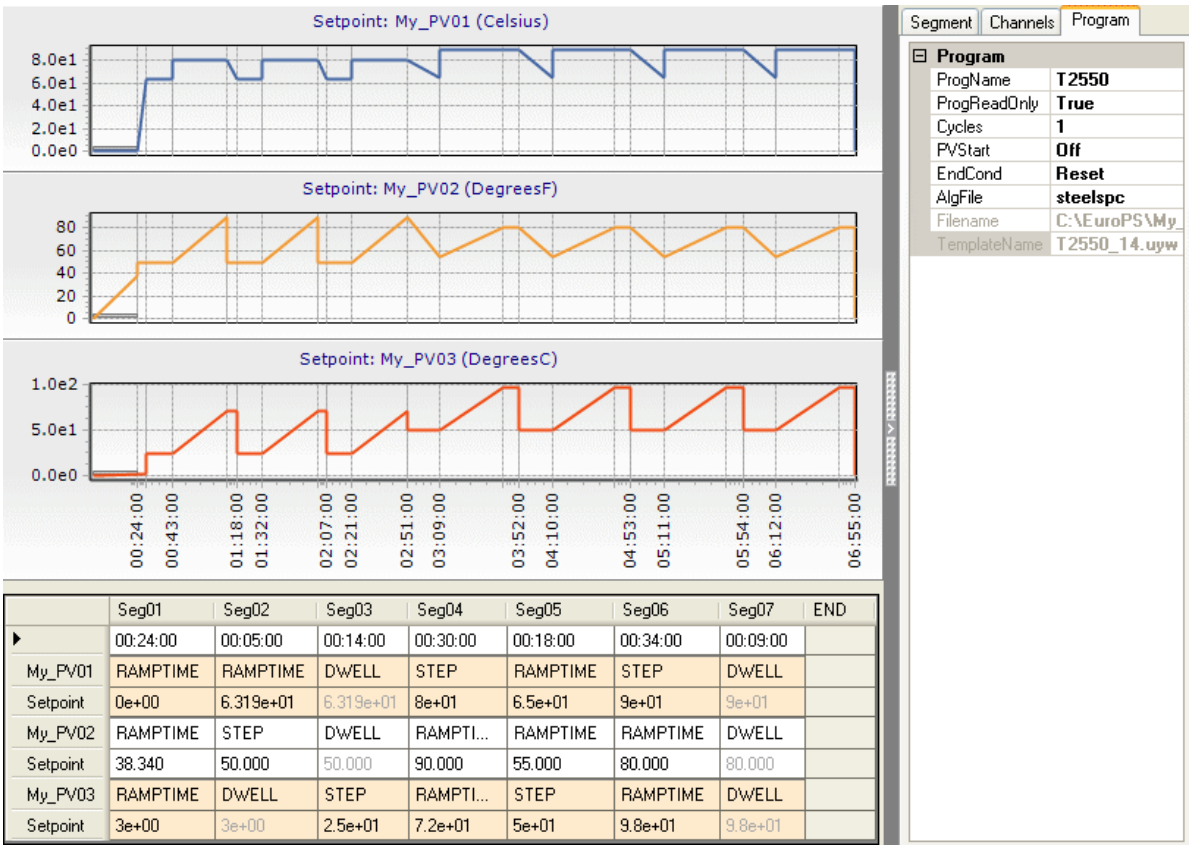
	Seg01	Seg02	Seg03	Seg04	Seg05	Seg06	Seg07	END
▶	00:03:00	00:15:00	00:24:00	00:30:00	00:18:00	00:34:00	00:09:00	
My_PV01	RAMPRATE	RAMPTIME	DWELL	RAMPTIME	RAMPRATE	STEP	DWELL	
Rate/min	12	---	---	---	0.8333332	---	---	
Setpoint	3.6e+01	6.319e+01	6.319e...	8e+01	6.5e+01	9e+01	9e+01	

16.3 MULTI-CHANNEL PROGRAM

A Multi-Channel Program is suitable for applications where a more than one Setpoint is profiled, e.g. environmental chambers that control temperature and humidity.

The Program type is pre-defined in a Program Template file that is generated by the Programmer Wizard in LINtools. A Multi-Channel Program creates a Profile for each Name defined in the Programmer Wizard.

Exit conditions can only be configured to occur in segments that are not moving the setpoint, i.e. Dwell and Step. All Profiled Channels must be Step or Dwell segments in order to meet this condition, see Seg03 below.



	Seg01	Seg02	Seg03	Seg04	Seg05	Seg06	Seg07	END
	00:24:00	00:05:00	00:14:00	00:30:00	00:18:00	00:34:00	00:09:00	
My_PV01	RAMPTIME	RAMPTIME	DWELL	STEP	RAMPTIME	STEP	DWELL	
Setpoint	0e+00	6.319e+01	6.319e+01	8e+01	6.5e+01	9e+01	9e+01	
My_PV02	RAMPTIME	STEP	DWELL	RAMPTI...	RAMPTIME	RAMPTIME	DWELL	
Setpoint	38.340	50.000	50.000	90.000	55.000	80.000	80.000	
My_PV03	RAMPTIME	DWELL	STEP	RAMPTI...	STEP	RAMPTIME	DWELL	
Setpoint	3e+00	3e+00	2.5e+01	7.2e+01	5e+01	9.8e+01	9.8e+01	

17 INTELLECTUAL PROPERTY RIGHT PROTECTION

This is the principle of preventing unauthorised access to or malicious use of configured strategy files. LINtools allows protect of Intellectual Property by password protection of files. Protected files can be opened only by users who know the password and can be used only on authorised instruments.

A protected file is encrypted using a key. This is a public identifier and a private password. A key must be specified to protect a file, but the same key can be used to protect multiple files. When opening a protected file a dialog is displayed showing the key identifier, and asking for the password. The protected file can be opened only if the correct password is used.

Caution

LINtools cannot open a protected file without the correct password. Protection and storage of passwords is the user's responsibility; lost passwords cannot be recovered nor can protected files be opened.

To use a protected file the instrument must be authorised with the correct key, as specified during instrument configuration. The same key can be used by many instruments. An instrument can be authorised with up to seven keys, which allows one instrument to use files protected with different passwords. A protected file cannot be used by an instrument that is not authorised with the correct key. This prevents users copying and using the strategy on different instruments without permission.

Note: The keys are managed using LINtools.

18 ALARM SUPPRESSION

18.1 OVERVIEW

For full details for implementing Alarm Suppression please refer to the 'Alarm Suppression User Guide', HA030272.

Alarm Suppression allows active alarm conditions to be suppressed for a period which can be configured between 0 and 90 days with a granularity of one second. This is achieved under strategy control via ST (Structured Text), Cold Start Parameter File (CPF) or via the operator interface of suitable instruments.

An alarm can be placed into suppression or removed from suppression by configuring the time duration of the alarm suppression attribute, '**.Suppress**'. Specifying a time of zero removes suppression, a time between 0 and 90 days sets suppression and, where allowed, a value of '-1' provides indefinite suppression.

Excluding the 'Combined' and 'Software' alarm, which can never be suppressed, Alarm Suppression can operate on each alarm independently or on the whole function block. Unsuppression of all 'local' LIN blocks in an instrument's database can be achieved via the database 'Header Block'. Alarm Suppression applies only to Alarms set with a priority between 1 and 15.

Note: Enabling Alarm Suppression impacts on database size. 'Alarm Suppression User Guide', HA030272 provides full details

18.2 SUPPORTED ALARMS

Alarm suppression cannot be applied to system alarms. Within LIN function blocks, there are only two true 'system alarms' - the 'Software' and 'Combined' alarms which are the first & last alarms in every function block type. All other alarms are considered to be 'Process Alarms', e.g. 'Absolute High Alarm'. Alarms referred to in this 'Help Guide' relating to Alarm Suppression are considered as 'Process Alarms' unless otherwise stated.

As mentioned above, Alarm Suppression can be applied to each individual alarm or to a (local) block as a whole. In the latter case, all alarms in the block are suppressed with a single time duration. Although effected as a single operation, it actually causes all alarms to be set individually, i.e. it has exactly the same effect as setting the same suppression duration time to each of the individual alarms contained within the block.

Note: The rarely used Function Block 'DREC_CHANNEL' has limited Alarm Suppression Functionality. 'Alarm Suppression User Guide', HA030272 provides full details.

18.3 CONFIGURATION

Alarm Suppression is configured by implementing Structured Text(ST) within an SFC (Sequential Flow Chart), the Cold Start Parameter File (CPF) or user interface, e.g. Eycon™ 10/20Visual Supervisor. An alarm can be placed into suppression or removed from suppression by configuring the time duration of the alarm suppression sub field attribute, '.Suppress'.

Specifying a time of zero removes suppression; a time between 0 and 90days sets suppression and a value of '-1' provides indefinite suppression. All 'standard' user interfaces allow entry of a time duration which must be a positive value. This means that 'standard' user interfaces cannot apply indefinite alarm suppression. This problem can be overcome by implementing a 'user screen' or 'mimic' strategy as discussed in the 'Alarm Suppression User Guide', HA030272.

18.3.1 Addressing and syntax

The alarm suppression attribute is addressed by the sub field attribute '.Suppress' for each individual alarm name or alarms in the whole block, and conforms to the ST syntax as follows:

Block.Alarms.AlmName.Suppress

Block.Alarms.Suppress

Examples:

- PID1.Alarms.HighAbs.Suppress=-1 - sets indefinite suppression for the 'PID1 Absolute High Alarm'
- PID1.Alarms.HighAbs.Suppress=0 - unsuppress the 'PID1 Absolute High Alarm'
- PID1.Alarms.Suppress=T#1d.12.2h - sets 36hrs suppression for 'PID1' All Process Alarms with priority 1-15.

The same syntax is used in all cases where access to alarm suppression is required as follows:

- Writing non-zero places the alarm into suppression for the specified duration in seconds. 'T#' format for specifying times can be used.
- Writing zero removes the alarm from suppression.
- Writing the value '-1' places the alarm into indefinite suppression.
- Reading the value zero indicates the alarm is not currently in suppression (individual alarms only).
- Reading positive non-zero values indicates the remaining time duration of suppression in seconds (individual alarms only).
- Reading the value -1 indicates the alarm is indefinitely suppressed (individual alarms only).

INVALID ENTRIES

The 'time duration until suppression is automatically removed' is specified in whole seconds and is limited to 90 days. Attempting to write a value greater than this clips it to 90 days. If specifying indefinite suppression, only the value '-1' is accepted, any other negative number is ignored. If multiple 'writes' are made to the suppression attribute, the last value written is used.

18.4 ALARM ACTION ON SUPPRESSION

When an alarm is suppressed the associated 'In_Alarm' bit is seen as 'FALSE' by 'strategy clients' listed as follows:

1. Local and cached function blocks wired within the Function Block Diagram (FBD).
2. As read by Structured Text (ST) within an 'Action Block' or SFC.
3. As viewed via block inspect using, e.g.Visual Supervisor FBM, EurothermSuite® point page, LINtools monitor, terminal configurator, etc.
4. Collection blocks such as 'GROUP' and 'ALC'.

18.5 FUNCTION BLOCK WIRING DIAGRAM (FBD)

FBD wiring access is not provided to the Alarm Suppression attributes.

Within the FBD the 'In_Alarm' attribute of an alarm is referenced as the source of a 'wire' using an identifier string of the form:

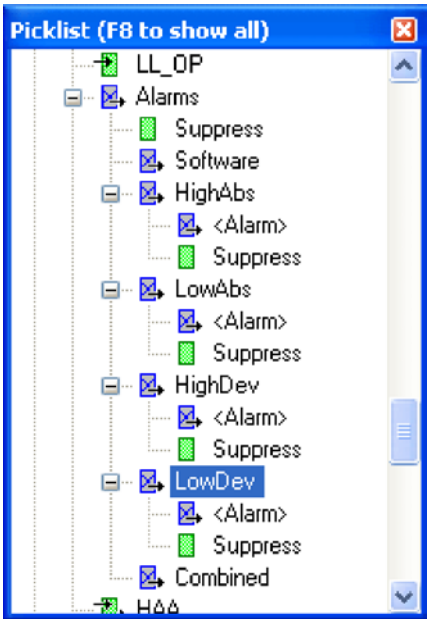
Block.Alarms.AlmName

18.6 STRUCTURED TEXT (ST)

When assigning a value, the 'T#' form of specifying times can be used, however a simple or derived numeric argument specifying seconds can also be used. When reading the value, it is always numeric in seconds.

18.6.1 LINtools Variable Pick-list

For configuring an 'action' within an SFC or Ladder diagram a pop-up Picklist or Palette Variable list is presented for use in the LINtools ST, SFC or Ladder diagram editor, all offering the alarm suppression attributes as shown below. The required 'Suppress' attribute must be selected first before placing it in the appropriate editor.



18.6.2 Written Statements

Structured Text(ST) can be written without the aid of a 'Picklist' where alarm suppression can be implemented using ST in the context of SFCs, 'user screen' actions and the 'Cold Start Parameter File'. ST applies alarm suppression using the syntax as shown above and for 'conditional' statements are applied as follows:

if(Block.Alarms.AlmName.Suppress > 0) THEN ..... conditional ST

Writing invalid or out-of-range values are treated as above.

18.6.3 Structured Text Versions and Errors

- If ST is included in the configuration of a database (e.g. SFC), there are three possible conditions in which ST 'errors' could occur and LINtools indicates appropriate warnings as follows:
- Usage of the '.Suppress' sub field attribute in ST and a SW version of LINtools that does not support alarm suppression results in 'Save' errors as shown below. For compatibility details refer to 'Alarm Suppression Supported Products'
- The target instrument's version of firmware is pre Alarm Suppression, see 'Alarm Suppression Supported Products'
- Alarm suppression is not enabled for the database being worked on, see 'Enable Alarm Suppression'

Saving...

Error in Action "Not_SupA" at text "[TIC-001.Alarms.Supp" : Can't find this name.

Error in Action "Suppress" at text "[TIC-001.Alarms.High" : Can't find this name.

Typical ST 'Save' Error Message

18.7 ALARM SUPPRESSION SUPPORTED PRODUCTS

Alarm Suppression is supported by a the following products and software. The upgrading of a database to support this function is detailed in 'Alarm Suppression User Guide', HA030272.

- T2750 Eurotherm PAC
- T2550 Version 7.0 Onwards
- Eycon™ 10/20 Visual Supervisor Version 5.0 onwards
- EurothermSuite®\NTSE Version 4.9 Onwards (with Wonderware® InTouch® 10 Shell Application)
- LINtools (Tactician) Version 4.9 Onwards
- User Screen Editor Version 2.5 Onwards
- T800 Not Supported
- T940X Not Supported
- T640 Not Supported

18.8 DOES THE EXISTING DATABASE SUPPORT ALARM SUPPRESSION?

Open the DBF using the appropriate version of LINtools and establish if an 'Alarm Suppression Enabled' checkbox is available under the 'Database' tab. If it is shown, this indicates that the DBF is already suitable for use with alarm suppression and is enabled by 'ticking' the 'Alarm Suppression Enabled' checkbox.

If the 'Alarm Suppression Enabled' checkbox and wording is not shown, the DBF does not support alarm suppression and requires upgrading. Refer to the 'Alarm Suppression User Guide', HA030272 for further details.

.

19 RAW COMMS

For full details of Raw Comms, please refer to the 'Raw Comms User Guide', HA030511

The RAW_COM Function Block provides the facility for LIN based products to control the transmission and reception of messages and protocols over a serial link and can also facilitate multi-node applications if required. It is available for applications where it is necessary to have low level control of the serial communications port to provide the flexibility to construct or analyse messages and protocols exactly as transmitted or received via a serial link.

The block includes the ability to execute structured text (ST) Actions. It performs basic functions first and then executes any ST Actions that have been created. The ST is stored in a file and is handled in the same way as for an 'Action' block and cannot access data outside the Raw Comms block. For protocols that are too complex to be handled using the ST inside the block an SFC should be used to drive the block which consequently provides considerable flexibility and is not limited by the constraints inherent in the block.

The RAW_COM block can be assigned to any serial port (if more than one port is available) and is designed for use by LIN instruments which support serial communications. It provides a wide range of low level facilities including:

- Direct access to messages as transmitted or received via the serial link.
- Independent control of message transmission and reception (limited to the same Baud rate).
- Selectable echoing of received characters when required.
- User selectable 'Delete' sequence for character deletion when required.
- The ability to be used in conjunction with SFCs for complex protocol support.
- Additional wide string variable blocks to assist in processing long byte and character sequences.
-

19.1 RAW COMMS SUPPORTED PRODUCTS

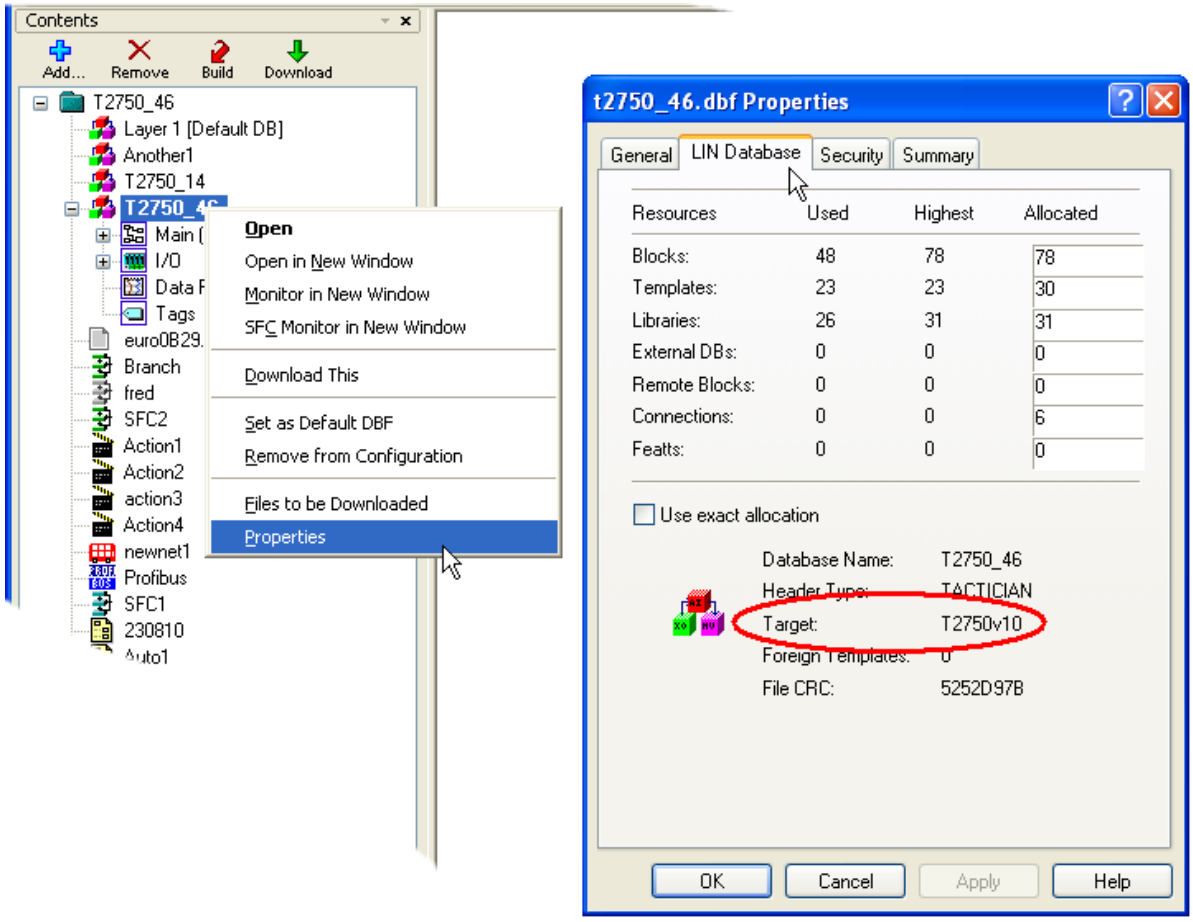
Raw Comms is supported by the following products and software. If the Instrument supports Raw Comms, but is a pre Raw Comms Version, the existing DBF does not support Raw Comms and must be upgraded (refer to the Raw Comms User Guide, HA030511, for details)..

- T2750 Eurotherm PAC all versions
- T2550 Version 7.0 onwards
- Eycon™ 10/20 Visual Supervisor Version 5.0 onwards
- EurothermSuite®\NTSE Version 4.9 onwards
- LINtools (Tactician) Version 4.9 onwards
- T800 Not Supported
- T940X Not Supported
- T640 Not Supported

19.2 DOES THE EXISTING DATABASE SUPPORT RAW COMMS?

Check the 'Instrument Version' that created the DBF as follows:

1. In the Contents pane, right click on the appropriate database symbol
2. In the context menu, select 'Properties'.
3. Click on the 'LIN Database' tab.
4. Check against the list above, that the 'Target' version is compatible with Raw Comms. support.



Database Properties Display

19.3 CONFIGURATION

This topic describes the setup decisions that need to be made prior to implementing a Raw Comms configuration, i.e. whether to use just an associated Action or just an associated SFC or an associated SFC interacting with an associated Action, as discussed below. A decision on whether just an associated Action can be used on its own without an associated SFC is determined by the required protocol complexity, bearing in mind that the maximum size of an associated Action file object code is 1000 bytes.

Raw Comms is configured in two or three parts as determined by the complexity of the required protocol. The RAW_COM block allows for execution of an associated Action which typically is used for non-complex protocols. For complex protocols an Action can also be used for 'common routine' tasks in association with an SFC. For further information to aid configuration please refer to the LIN Blocks Reference Manual, and to the Raw Comms User Guide, HA030511 for implementation

examples. For reference to Simple Variable Application Blocks to assist in Raw Comms processing also refer to Chapter 9 of the Application & Control Modules Manual.

19.3.1 Configuration of the RAW_COM function block fields

The RAW_COM block fields are configured prior to any associated Action or SFC creation. Refer to the LIN Blocks Reference Manual, HA082375U003, issue 18 onwards

Note: On Instruments where multiple user tasks are supported the RAW_COM block can be allocated to run on any task.

19.3.2 Associated Structured Text (ST) Action

For non-complex protocol applications an associated Structured Text (ST) Action can be used to facilitate the required protocol processing. This requires that two fields are configured in the RAW_COM block. These fields are configured with the associated Action itself and the Filename containing the Action and are ActName and FileName respectively. The procedure to create an ACTION is described in the Raw Comms User Guide, HA030511. To force the RAW_COM block to load a new associated Action refer to Raw Comms on-line reconfiguration.

19.3.3 Associated SFC

For more complex protocols an SFC will be required to manage the overall protocol state processing requirements. However in this case it may be possible to process 'common routine' tasks using an associated Action which interacts with an SFC. The SFC manages the overall protocol 'state processing' tasks.

An associated SFC interacting with the RAW_COM block is considered just like any other SFC and the process for creating and implementing the SFC is no different.

19.3.4 Also See

For further information relating to ST usage available to assist in Raw Comms processing also see:

ST Constants

Operators and Functions

19.4 ON-LINE RECONFIGURATION

On instruments that support on-line reconfiguration (e.g. T2550 PAC), it is possible to create, remove or replace a RAW_COM block as required. However, this will be successful only if the serial port protocol has already been configured to 'Raw', via the Instrument Options Editor, refer to Raw Comms User Guide, HA030511.

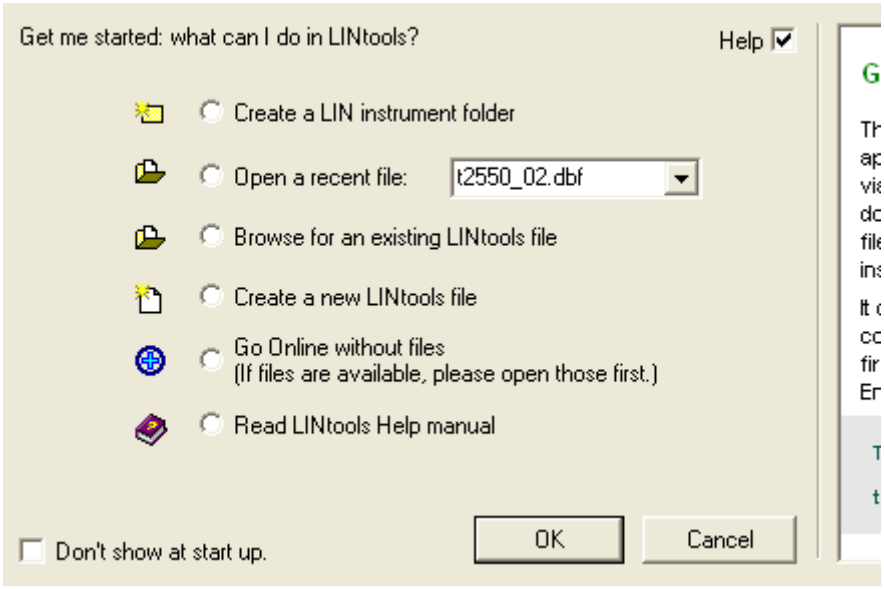
If an associated action file has been specified for the created or edited block, when in ‘TRY’ mode this file is read and reloaded. This method can therefore be used to achieve an associated Action reload, which is not possible while under normal running conditions.

20 HOW TO...

20.1 USE LINTOOLS

20.1.1 Use the Get Me Started wizard

This wizard offers the most common operations required when first opening LINtools. By default, it is displayed on starting the LINtools, but this can be disabled by setting the 'Don't show at start up' checkbox ☑.



Note: Context help is displayed when the 'Help' checkbox is set ☑.

To initiate any of the options displayed,

1. Click on the relevant radio button.
2. Progress through the wizard until all appropriate fields are complete, when the Finish button appears.
3. Click on the Finish button to finalise the operation, and automatically Build the selected options.

20.1.2 Add a new network/instrument

These commands are used to create additional Modbus or Profibus sub-networks and then, with a separate, Add New Instrument command, create an instrument on a network.

To add a new network,

1. In the **Contents** pane, select a LIN instrument name to reveal a context menu, and select **Add New Network…** to reveal the New Network Wizard.
2. Complete the each field, including network type, Modbus or Profibus, and instrument I/O and OPC Port data, pressing **Next** to continue to the next page of the wizard. Information relating to a selected field is displayed when Help is set ☑.
3. Finally, press **Finish** to accept the new network configuration.

Note: You will now be aware that the network contains an instrument indicated by the '⊞'. Using the Explorer command will open a window to display the new network.

To add a new instrument,

1. select the **File** > **Get Me Started** command to display the Get Me Started dialog offering the most common operations used when initially starting LINtools Engineering Studio

   - Alternatively, in the **Contents** pane, select a Network name to reveal a context menu, and select **Add Instrument…** to reveal the New Instrument Wizard.

2. Complete the each field, including instrument name, instrument type and version, Clone file Template, and Slave address, pressing **Next** to continue to the next page of the wizard. Information relating to a selected field is displayed when Help is set  .

3. Finally, press **Finish** to accept the new instrument configuration.

Note: You will now be aware that the network contains an instrument indicated by the '⊞'. Using the Explorer command will open a window to display the new network.

20.1.3 Create a new LIN Instrument Folder

A new LIN Instrument folder will be created to represent a specific Instrument on the 'live' Network. It is simply a way of retaining any file in single location. Many of the files will be used to configure an instrument, but an Instrument Folder can contain any files that may, or may not be relevant to this instrument. It will be appear in a specific Project Folder in defined Network using the New LIN Instrument Folder command.

This command launches the New LIN Instrument Folder wizard allowing you to create a new Instrument Folder in new or existing Projects, and new or existing Networks.

To create a new LIN Instrument folder

1. Select **File** > **New LIN Instrument Folder** command to launch the New LIN Instrument Folder wizard.

2. Complete each page of the wizard as required using the dynamic help displayed to the right of the dialog when necessary.

Note All Instrument Folder specifics entered here can be inspected at any time via the <Instrument Type> Instrument page in the Instrument Folder Properties dialog. However, changing these parameters will NOT create default Instrument Folder files.

3. Press Finish to complete the wizard.

   This will automatically create a LIN Instrument Folder and a Configuration (Header) block corresponding to the Instrument type and version indicated using this wizard.

20.1.4 Open the Editor window

A blank LINtools Engineering Studio window allows any LINtools related operations to be initiated. This includes creating and editing LINtools files, e.g. **Databases**, **Sequences**, and **Actions**, creating a new Project Folder and subsequent Network Folders, and even opening any Tools that are related to the operation of the LINtools file, i.e. Setpoint Programmer, User Screen Editor, iTools Engineering Studio and Modbus tools.

Note: Each related Tool can be accessed via the Tools option in the Menu bar.

From this window you can access any of the editors.

To open a blank configuration window:

1. Select Start >All Programs > … > LINtools Engineering Studio.

Note: '…' denotes the installation path.

- The Get me started wizard is displayed if operated for the first time or the '**Don't show at start up**' checkbox is not set 

This wizard tool offers the most common commands used in LINtools Engineering Studio, and can be launched a any time.

- A blank worksheet appears if the '**Don't show at start up**' checkbox in the wizard is set 

20.1.5 To edit an existing configuration:

1. In LINtools, click the Open toolbutton (or use **File** > **Open…**) to display an Open dialog.
   - Alternatively, use the **Get me started wizard**.

   The 'Open a recent file' offers a list of recently saved files, any of which will be opened after selecting this radio button. Other radio buttons offer the option to create or open another Instrument Folder.

   If you have opened this file recently, you can click it directly, **File** > **Recent Files list**.
2. In the dialogue, select a file and click Open. The corresponding editor opens, ready for you to start the configuration.
   - Alternatively, simply double-click the file icon in the Project instrument folder.

Note: Any currently unsaved LINtools configuration file MUST be saved before attempting to open another configuration file.

20.1.6 Open a file in a New Window

This allows a file from another application to be opened.

Note: If a new LINtools window is opened, it allows differences between download lists to be checked and allows function blocks to be 'dragged' from one window to another, creating cached function blocks.

To open a file in a new window,

1. In the Contents pane, highlight the required file name, e.g.. T800_04 [FBD], actset01 [SFC], <filename>.doc or <filename>.bmp.
   - It may be necessary to expand the networks (by clicking on the '⊞' icon) in order to locate the required file.
2. Right-click on the required file name to display a context menu.
3. Click 'Open in New Window' to open and display the selected file in the appropriate application window. (If this new file is a LINtools file, this will be the current LINtools window.)

Note: If the file has been password protected, it will be necessary to enter the correct password before the file can be opened.

20.1.7 Open a file in the current Window

It may be required to open a LINtools file in the current LINtools window.

1. In the **Contents** pane, highlight the required file name, e.g.. T800_04 [FBD], or actset01 [SFC].
   - It may be necessary to expand the networks (by clicking on the '⊞' icon) in order to locate the required file
2. Right-click on the required file name to display a context menu.
3. Click 'Open' to display the selected file in the current LINtools window

Note: If the file has been password protected, it will be necessary to enter the correct password before the file can be opened.

20.1.8 Import a file

It may be convenient to import a file from another folder into either of the Instrument folders, thus saving time-consuming repetition when the required configuration is similar to an existing one.

To import a configuration from another configuration,

1. In the Contents pane, highlight the folder name for the destination of the imported file, i.e. Instrument, if you wish to download the file to the instrument, or the Unused Files, for documents, and files not for downloading, etc..
2. Select the Instrument folder name to display the context menu.
3. Click **Import File…** to reveal an **Import File** dialog appears, which lists files of the same type as the current file.
4. Navigate to the file you want to import and click **Open**.

   The imported file is copied to the selected folder and immediately opened. If another worksheet is already open the user will be requested to save it before the continuing with the import operation.

Note: Imported Files having the same names as Files in the current Instrument folder displays an error message and are not imported.

20.1.9 Add to Configuration

This is available only from the Unused Files folder. If a file is now needed for downloading, it can be put in the Instrument folder using the Add to Configuration command via the context menu on the Unused Files folder.

To add a file,

1. In the **Contents** pane, show and open the Unused Files folder. Highlight the required file, i.e. <filename> [FBD], <filename> [SFC].
2. Select the highlighted file name by right-clicking on the highlighted file name to display a menu.
3. Click **Add to Configuration** command to move the selected file into the Instrument folder.

Alternatively, the file can be dragged from the Unused Files folder and dropped in the Instrument folder.

Note: This process can also be initiated via the ‘Add a file to the configuration button’ in the Contents Toolbar.

20.1.10 To view the Contents pane:

If the **Contents** pane is not visible, select **View** > **Contents** from the menu bar. It can be hidden using the same command, or by pressing the ✕ button at the top right hand corner of the Contents pane.

Note: Each Pane can be shown in either a minimised and maximised position using the ▾/▴ buttons respectively.

20.1.11 To view the Object Properties pane

During offline configuration the **Object properties pane** displays the characteristics of a selected item. However, when displayed during Online Reconfiguration, it shows the online state of a selected item, such as an added block, or a deleted wire.

Note: Each online state is indicated by symbol, colour, and line style.

In an **Object properties pane**, the operator can view the properties of a:

- LIN Sequence STEP or TRANSITION.
- LIN Database function BLOCK, and addition BLOCK pages, WIRE, or COMPOUND.
- Update Rate field
- Task field

To view an Object's properties pane:

- double-click the object in the sub-window (or single-click it if a properties pane is already open), or
- highlight the object and click the Properties toolbutton. or
- highlight the object, then in the **View** menu, select **Object Properties.**

The ▾/▴ symbol can be used to minimise or maximise the Pane if more than one pane is docked in the same area of the application, and the ✕ can be used to hide this pane.

20.1.12 Show/hide the Report pane

During offline configuration of any LINtools file, the **Report window** (Results pane) displays any errors that may have occurred when building the instrument strategy. This pane allows the inspection of any reported errors.

Note: Error codes are displayed in the Report pane as Hex values. A description of the error codes can be shown using the Error lookup utility available when selecting Start > Programs > ... > Utilities > Error lookup

To show/hide the Report pane

- Select the **View** menu, and then **Report Window.**

The ▾/▴ symbol can be used to minimise or maximise the Pane if more than one pane is docked in the same area of the application, and the ✕ can be used to hide this pane.

Note: The Results information will also appear in a new Build window after pressing the Show Build window button when building the instrument folder.

20.1.13 To show/hide the Status bar

During offline configuration of any LINtools file, the **Status bar** displays specific Editor information. However, when displayed during Online Reconfiguration, it can also show the online state of a selected item, such as an added block, or a deleted wire.

To show/hide the Status bar below the **Object Properties pane**, and/or **Report pane**

- Select the **View** menu, and then **Status Bar.**

Note: This also applies to the Contents pane, Object Properties Pane, Palette pane, and Report window.

FUNCTION BLOCK DIAGRAM (FBD AND FBD MONITOR) STATUS BAR

While editing and monitoring an FBD, the Status bar shows,

- Tag information
- the X, Y co-ordinates of the cursor in the active FBD window
- the % (percentage) view (zoom in or zoom out) of the active FBD window
- the particular Editor or Monitor in use (Database Editor or Database Monitor)
- the current LINtools operating mode, e.g. connected to 'Live' instrument, green led illuminated, or not connected to 'Live' instrument, green led extinguished

SEQUENTIAL FUNCTION CHART (SFC) STATUS BAR

While editing a Specific or Generic SFC, the Status bar shows,

- **DB:** The target LIN Database filename (if loaded)
- **Map:** The current Map filename (if loaded)
- the **X, Y** co-ordinates of the cursor if a chart Action window is active
- the **Ln, Col** (line and column) position of the text cursor if a Structured Text Action window is active
- the **%** (percentage) view (Zoom in or zoom out) if a chart Action window is active
- the particular Editor or Monitor in use (Sequence Editor or Sequence Monitor)
- the current LINtools operating mode, e.g. connected to Live' instrument, green led illuminated, or not connected to 'Live' instrument, green led extinguished.

SEQUENTIAL FUNCTION CHART (SFC MONITOR) STATUS BAR

While monitoring an SFC, the Status bar shows,

- **Seq <...>:** The filename of the remote Sequence.
- the **X, Y** co-ordinates of the cursor if a chart Action window is active.
- the **Ln, Col** (line and column) position of the text cursor if a text action window is active.
- the particular Editor or Monitor in use (Database Editor or Database Monitor)
- the **%** (percentage) view (Zoom in or zoom out) if a chart Action window is active.

ACTION EDITOR STATUS BAR

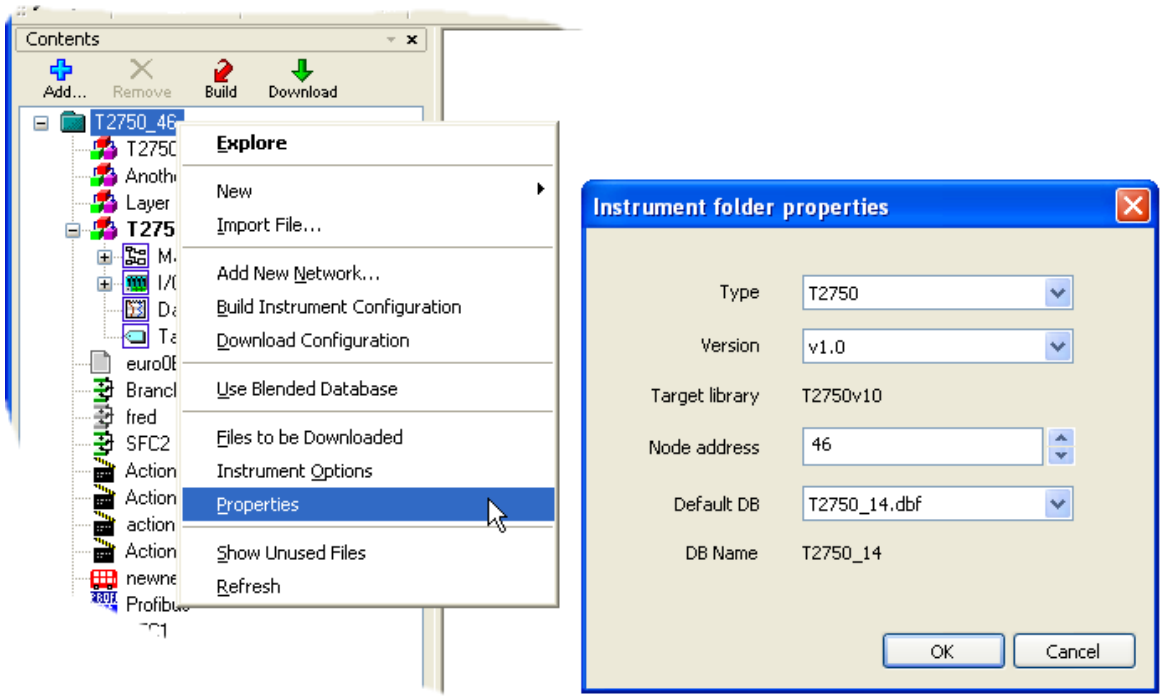
While editing an Action, the Status bar shows,

- the **Ln, Col** (line and column) position of the text cursor in the current LIN Action window.

20.1.14 To show the instrument properties

1. In the **Contents** pane, right click on the relevant instrument node name.
   - The networks may have to be expanded (by clicking on the '+' icon), to locate the required instrument.
2. In the context menu that appears, select 'Properties' to reveal the Instrument Properties display.

Notes:
1. Any changes made to fields in the Properties dialog may require additional configuration using other software applications which are not part of LINtools.
2. For some software versions 'Instrument Properties' is a tab ('LIN Instrument') in the 'Properties' display.



20.1.15 Open the Explorer

This displays an explorer window at the selected instrument and allows you to browse for files stored on local or networked computers.

To open the Explorer,

1. In the Contents pane, highlight the top level node name, e.g. T800_04.
2. Right-click on the highlighted file name to display a menu.
3. Click 'Explorer' to reveal the explorer window.

CONFIGURE THE INSTRUMENT PROPERTIES

This command is used to display the properties of the selected instrument. It reveals the information entered when using the New LIN (or Modbus) Instrument Wizard.

The '**Upload current…**' button initiates a wizard, that gathers specific parameters associated with an Instrument on the defined Network and LIN Node address.

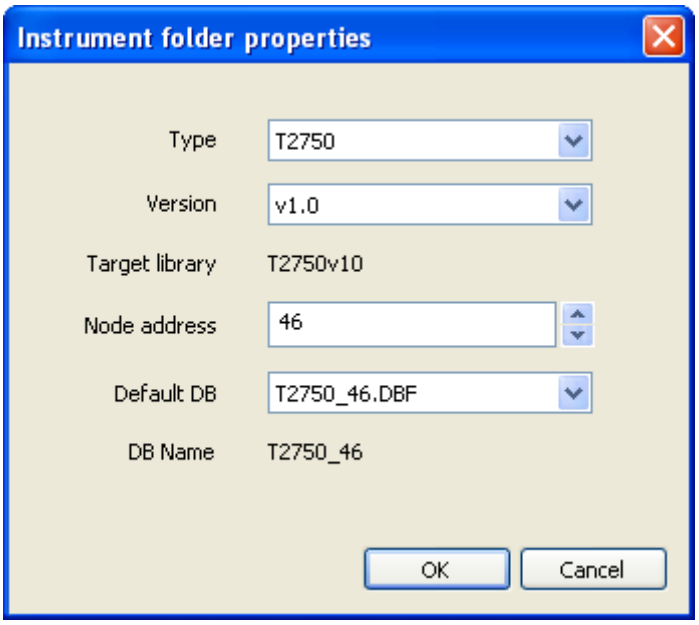
To show the instrument properties,

1. In the **Contents** pane, highlight the Instrument Folder name.
   - You may have to expand the networks, click the '⊞', to locate the instrument required.
2. Select the highlighted file name by right-clicking on it to display a menu.
3. Click **Properties** to reveal the LIN (or Modbus) Instrument Properties page.
   - Click the LIN Instrument page to reveal the unique LIN instrument data entered when using the New LIN (or Modbus) Instrument Wizard.

Note: Any changes made to fields in the Properties page may require additional configuration using other software applications which are not part of LINtools.

INSTRUMENT PROPERTIES - EXAMPLE

Note: Instrument properties fields are initially configured using the New LIN Instrument Folder wizard from within the Project Folder.



20.1.16 To download a configuration:

TOP LEVEL CONFIGURATION

Click on the Download tool icon 

at the top of the Contents pane, or:

1. In the Contents pane, right click the top level node name (e.g. T800_04) to reveal a context menu.
2. Click on 'Download Configuration' and confirm.

In either case, 'Download Configuration' attempts to download the files contained within the Instrument folder of the target instrument (specified in the 'Files to be Downloaded' list).

Confirming the download causes further Confirmation prompts to appear stating that instrument files will be overwritten, followed by a request to load and run the latest downloaded Strategy.

OTHER FILES

1. In the Contents pane, right click on the relevant file to reveal a context menu.
2. Click on 'Download This' and confirm.

'Download This' attempts to download the selected file to the target instrument. Confirmation is followed by a request to load and run the newly downloaded Strategy.

TO EDIT THE LIST OF LINTOOLS' DOWNLOADABLE FILES:

1. In the Contents pane, right click on the required file name (e.g. T2750_46, T800_04 [SFC], etc.).
2. In the context menu that appears, click on 'Files to be Downloaded'.
3. In the list that appears, a number of files are ticked. Most of these are essential files which cannot be 'unticked'. Select (click on the tickbox) any additional files to be downloaded to the instrument. Ticked files are downloaded to the instrument when a download is initiated is initiated.

20.1.17 Close a LINtools file

It may be required to close only the current LINtools application file, i.e. .dbf, .sdb, .stx, .udm, .uqd, or .upm. This can save time as LINtools remains open allowing another LINtools file to be opened.

To close a file,

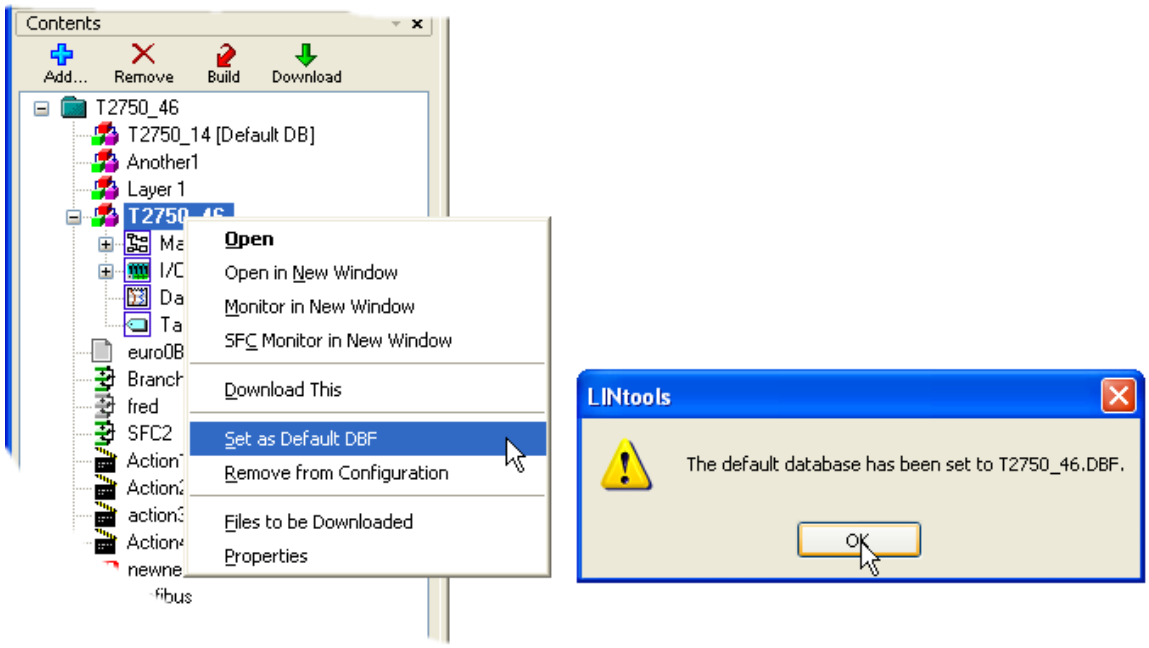
1. Select **File > Close**

- Save the file as required.

20.2 CONFIGURE DATABASES

20.2.1 To change the instrument's Default .dbf:

1. In the Contents pane, right click on the required database (<filename> [FBD]).
2. In the context menu, select 'Set as Default DBF' and accept the new configuration by clicking on 'OK'. (The command is unavailable (greyed out) if the selected file is already specified as the DefaultDBF.)



20.2.2 Create a LIN Database file

The best place to start creation of a Database file is within LINtools, using the '**Get me started**' wizard, as it allows the Project and Network to be created as well. However, it can also be created using the New LINtools file toolbutton or in the Project environment. That is, in the Explorer view of the Project file, inside the instrument folder that will run the Database.

Note: A new LIN Database file can be created elsewhere - e.g. on the desktop - but the file will need additional configuration to tell it the instrument type, etc.

To create the new LIN Database file

These instructions pertain to all other LINtools application files.

1. Select the top level instrument file and right-click to reveal a context menu.
   - Use the New LINtools file toolbutton.
   - Use the 'Get me started' wizard.
2. Select New > LIN Database. An Add new file to Configuration dialog appears.
3. Enter a concise filename. The extension .dbf is added automatically.

4. Another dialog appears requesting the Target Instrument Type. Select the Instrument and version type required and accept by pressing **OK**.
5. When the file is open, edit the LIN Database file, as required.
6. Finally, create or edit the Tags in the Tag table.

Note: Use Copy, Copy Block Parameters, Paste and Paste Block Parameters commands to reduce syntax error occurrences.

20.2.3 Access the LIN Database Editor

To access the LIN Database Editor:

1. Double-click the LIN Database file or its icon to launch LIN Database Editor.

Note: An automatically created LIN Database (_auto.dbf) may already exist in the instrument. This can be edited using the Online Reconfiguration toolbutton.

2. A **Select Target Library** dialog may appear, asking you to select a target instrument from the list.
   - If you choose a target and hit **OK**, only block templates that can run in the selected instrument will be offered initially for configuration.
   - If you **Cancel** the dialog, you will normally cause the database load to be aborted.
3. The Database Editor window now appears, displaying the **Contents pane** (empty for a new database). The Worksheet and Contents tabs, at the left of the window, list this Database as '**Main (ROOT)**, I/O (I/O configuration including appropriate slots), and Tags.

20.2.4 Create a new Layer

A single complete Strategy is generally created using a number of blended layers. Each layer of a Database file appears in a hierarchical style in the Configuration folder, which can be ordered appropriately.

 Order the Blended Database

This command allows you to create a new layer for the Database file of the selected node if the Use Blended Database command is selected, indicated by the tick.

Note: The required PROGRAM, or instrument specific, header block is automatically created during the build process.

To add a new layer,

1. In the **Contents** pane, open the Instrument folder. Highlight the required Database file, i.e. <filename> [FBD].
2. Select the highlighted file by right-clicking on the highlighted file name to reveal a menu.
3. Select **Add New Layer**…. The **Add new file to Configuration** dialog appears.
4. Enter a concise filename. The extension **.dbf** is added automatically.

Note: A request for Target Instrument Type may appear for if the parent instrument folder could not be found. Select the Instrument and version type required, and accept by pressing OK.

5. When the file is open, edit the Database file as required.

20.2.5 Change to and from a Blended Database

You may want to know which LIN Database configuration the selected LIN instrument currently is, and then subsequently change the LIN Database configuration between Standard Database and Blended (or Layered) Database configurations.

To change the Database configuration,

1. In the **Contents** pane, highlight the Instrument Folder, node name, i.e. T800_04.
2. Select the Use Blended Database command from the context menu.

If a Blended Database is in use the command is set  .

- Click **Use Blended Database** command to change the Database configuration.

Note: Existing layers are automatically removed from the configuration and held in the Unused Files folder.

If changing to a Blended Database configuration, either

- the build list (build.ubl.old) is converted (to buildlst.ubl) and restored, re-establishing the Blended Database configuration.
- if no backup (build.ubl.old) exists, a new build list (buildlst.ubl) is created with the Default DBF installed as the base layer.

If changing to a Standard Database configuration, the layers of the existing Blended Database are separated and moved as individual files to the Unused Files folder. The Default DBF converts to a Standard Read/Write Database file (.dbf).

Note: A backup of the Blended Database's configuration is automatically created by converting and renaming the buildlst.ubl to build.ubl.old. Therefore this will not be recognized as a Build file.

20.2.6 Ordering a Blended Database

A single complete Strategy is generally created using a number of blended layers. Each layer (.dbf or .ujc) of a Strategy is indicated in a hierarchical style in the Instrument Folder, which can be ordered appropriately.

Note: The DefaultDBF and Auto generated layers are all Read Only files and do NOT require a graphics file, but each layer MUST contain a PROGRAM (Header) block with the same name as the DefaultDBF.

To order a Blended Database,

1. In the **Contents** pane, highlight the required file name, e.g.. T800_04 [FBD], or actset01 [SFC].
   - It may be necessary to expand the file structure (by clicking on the '+' icon), in order to locate the required file.
2. Select the highlighted layer name and drag it to the layer it should follow.
   - The user may also select a file from another LIN Database file and drag it to the layer it should follow.

Note: Auto generated layers are blended immediately after the Base Layer only.

3. Drop when the preceding layer is highlighted.

20.2.7 To import a Layer

Used to add an existing LIN Database layer from an existing LIN Instrument to this LIN Database

1. In the Contents pane, open the Configuration folder. Right click on the required Database file, i.e. <filename> [FBD].
2. Select 'Import layer...' An **Open** dialogue appears, which lists files of the same type as the current file.
3. In the browser window that appears, navigate to the file to be imported and click 'Open'.

The imported file is copied to the selected folder and immediately opened. If another worksheet is already open the user is asked to save it before the import operation continues.

Note: Any attempt to import a file which has the same name as a file already existing in the instrument folder will fail and an error message will be generated.

20.2.8 configure a PID Control Loop Strategy - Tutorial

To provide a basic introduction to what LINtools can achieve, use the following instructions to configure a PID control loop strategy for a Visual Supervisor instrument, used for recording and visualization, connected to an I/O instrument, used to control inputs and outputs in a plant/system.

The PID control loop is actually created in the two instruments, meaning that some blocks must be cached.

To configure a PID control loop strategy,

1. Place the Function blocks.

   Placing blocks on the worksheet defines how the information will be processed through the strategy.
2. Wire the Functions blocks.

   Wiring the blocks together defines the information flow through the strategy.

Note: Use the filtering icons and drop-down menu on the picklist to ensure only the required data types are available when configuring Variable Line tab parameters.

3. Edit the Function block fields.

   Editing the required block fields defines the limits applicable to the Strategy, that can be used to identify and help diagnose problems that may occur in the strategy.
4. Test the Strategy.

   Testing the Strategy ensures that the PID control loop operates correctly.

20.2.9 Work with function blocks

PLACE A FUNCTION BLOCK

To place a function block on the current worksheet:

1. Click the function Palette toolbutton  at the top of the window, or click on the 'Palette' item in the Views menu.
2. In the palette, check that the **Library** is set to the appropriate target instrument. If necessary, select another target from the pulldown **Library** menu, e.g. 'T940v1.1'. A list of Header blocks supported by this target appears in the palette. Each header block displays a 'head' category symbol.

Note: With an empty worksheet, you cannot select a non-header function block.

There are two types of **Header** block (CONFIG blocks):

- a Regular header block.
- a Program header block.

3. Click the required header block in the list, e.g. 'TACTICIAN' for a T2550, to see help information on the block in the help pane below the template palette.

4. To place the header block:

   i. Drag it onto the worksheet, or with it highlighted,
   ii. Double-click it to load the cursor with the function block, then position the loaded cursor on the worksheet and click to paste the function block down, or
   iii. Click on the Block icon () to load the cursor with the function block, then position the loaded cursor on the worksheet and click to paste the function block down, or
   iv. Select 'Block' in the 'Make' menu to load the cursor with the function block, then position the loaded cursor on the worksheet and click to paste the function block down.

Notes:
1 An invisible grid determines the exact position of the pasted function block.

2. To delete a function block, click on it once to highlight it, then click the PC's <Delete> key. To delete several function blocks, multiple-select them, then Delete as above.

5. To place other function blocks, click a function block category from the list at the left of the palette. A list of function blocks in that category appears at the right of the palette.

Note: A Strategy for the T2550 (Tactician) instrument must include one MOD_UIO block per I/O Module fitted in the instrument.

6. Place your chosen function block by dragging or by loading the cursor, as described in step 4.

Note: To place more than one instance of a function block via the loaded cursor, hold down the computer's <Ctrl> key as you paste the function block down. This leaves the cursor still loaded and ready to paste more instances. Right-click the mouse to escape from this mode.

NAME A FUNCTION BLOCK

To name a function block or group of blocks:

1. Start by highlighting the function block or group of function blocks required to be renamed.

2. Select the **Rename…** command.

   - From the Editor window **Edit** menu, select **Rename...**.
   - From the FBD sub-window, right-click on the function block (or any one of the selected function blocks) to reveal a context menu. Select the **Rename** item to display a **Rename dialog**.
   - From the Find… dialog, Block tab, right-click on the function block and select the **Rename** item to display a **Rename dialog**.

3. If only a single block is to be renamed, type in a new function block name and click **OK**. Alternatively, select a function block naming pattern from the pulldown menu in the dialog, then click **OK**.

   - If you are renaming a group of function blocks a suitable renaming pattern should be entered for the whole group.

Note: The renaming process does not allow invalid function block names to be entered. If an error is detected the user is asked to select an alternative name/pattern.

USE THE FIND UTILITY

In a large configuration it can be difficult to find a particular function block, text string, action, step, etc. This utility helps to locate such objects, as follows:

1. Click the 'Find...' item in the Edit menu. The 'Find...' display appears, containing one or more tabs, depending on the context.

2. Click a tab to display the required page:

   - a Block page - finds function blocks.
   - a Field page - finds function block fields.
   - a Text page - finds text strings in Actions, comments, and ladder diagram fields.
   - an Action page - finds chart, ladder, and text Actions.
   - an Association page - finds Action associations.
   - a Step page - finds Sequence Steps.
   - a Tag page - finds Tags.
   - a Changes page - finds changes between the Strategy files on the Computer and the 'live' Instrument.

Note: Some pages contain buttons that are identical to the commands available on the context menu.

HIGHLIGHT A GROUP OF FUNCTION BLOCKS

A group of function blocks can be selected (highlighted) in the following ways:

- <**Ctrl**> key selection

  Click on the first function block in the group to highlight it. Then, whilst holding the <Ctrl> key down, click further function blocks to add them to the multiple selection.

- Ring Selection

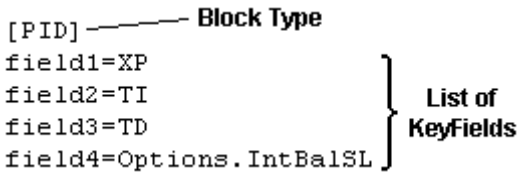
  Position the cursor at one corner of an imaginary box surrounding the function blocks to be selected. Holding down the left mouse button, drag out the box to enclose all the function blocks. Release the button to highlight the function blocks.

INSPECT FIELDS IN A FUNCTION BLOCK

Double-click anywhere within the function block symbol (except over the template icon) to open the **Object properties pane** at the foot of the worksheet. The white areas of the function block specification may be available for editing.

Note: You only need click the function block once if a pane is already open at the foot of the worksheet.

- Specified Key field parameters of offline instruments only are displayed on a tooltip if the KeyFields toolbutton is pressed or **View** > **Show Key Fields** is selected.



- Existing Comment tab Text is displayed on a tooltip if the **Show Comment** Tooltip is enabled on the **Editor** tab in the **View** > **Options** > **Settings** dialog.

Note: A Comment tab Text symbol in the upper right corner indicates that Comments are present.

INSPECT WIRE INPUTS TO A FIELD

Inputs wired to fields are shown in the **Block tab** by an arrow → to the left of the field name. Inputs to bitfields are shown with a double arrow ⇉.

1. To trace the source of the input, right-click the arrow to display a menu.
2. Click the **Goto Wire Source** item. The function block sourcing the wire highlights and relocates to the centre of the worksheet. The **Block tab** now displays the fields of the source function block.

EDIT FUNCTION BLOCK FIELDS

1. In the Object Properties Pane, select the 'Block' page of the required function block.
2. For numeric and string fields, click once in the field to highlight it, then type in a new value to completely overwrite the old one. Alternatively, double-click the field to insert a text cursor and use this to carry out additions or deletions to the current value. Enter the new value by operating the computer's <Enter> or 'Return' key.

Note: Double-click a field containing a text cursor to highlight all characters in the field, for replacement.

3. For menu fields (e.g. **Mode**), double-click the field to highlight it then pull down the menu of options. Select an item then hit **<Return>** to enter it.

Note: With a menu field highlighted, type in the initial letter of the required option to select it quickly. If several options have the same initial letter, typing the letter repeatedly cycles the selection round these options.

4. For bitfields and Alarm fields, click once to reveal a pane to the right of the function block specification pane, in which current values can be edited. Operate <Enter> or 'Return' after each individual bit has been edited.
5. For the 'DBase' field, double-click it to display a DBase dialog in which a remote function block's database and node address can be edited. Enter the new value by clicking on 'OK'.

To get help for an individual function block field:

Click on a field name or value and press <**F1**> to reveal a help window for that field.

EDIT A FUNCTION BLOCK'S DATABASE

1. Highlight the function block or group of function blocks to be edited.
2. Right-click on any one of the selected function blocks to reveal a context menu. Select the Database item to display a DBase dialog.
3. Fill in the dialog, then click OK to enter the new data.

A cached function block displays a  symbol in its lower left corner.

You can create a cached function block directly using drag-&-drop from another FBD.

EDIT A FUNCTION BLOCK'S UPDATE RATE

1. Highlight the function block or group of function blocks required to be edited.
2. Right-click on any selected block to reveal a context menu. Select the Update Rate item to display a Rate dialog.
3. Fill in the dialog, then click OK to enter the new data.

INSPECT A FUNCTION BLOCK EXECUTION ORDER

Click the Order toolbutton to display the function block execution order (if available) as a set of numbered blue boxes attached to the top right corner of each function block symbol. The numbering starts from '1'.

In instruments with several 'tasks', each Task has a separate and independent uniquely-coloured series of execution order numbers, each starting from '1'.

- Grey-coloured numbers denote function blocks that have not yet been allocated to a Task, or that cannot for other reasons show a meaningful execution order, e.g. cached function blocks.

CREATE A CACHED FUNCTION BLOCK

The purpose of using cached blocks is to share one instrument's data with another.

Note: It is recommended that wires are connected from the block in the Local instrument database file to the cached block.

To cache a function block from a Remote instrument database file:

1. Open the Local instrument database file.
2. Open the Remote instrument database file.
3. In the Remote instrument database file, select the block, or several blocks, to be cached in the Local instrument database file.
4. Drag these blocks from the Remote instrument database file, and place them in the Local instrument database file. The block(s) are now cached as indicated by the cached symbol.

Note: If the Local instrument database file is hidden on the desktop, drag the block(s) from the Remote instrument database file to hover over the Local instrument database file in the taskbar. The Local instrument database file will appear, allowing you to place the block(s) from the Remote instrument database file.

COMPOUND A GROUP OF FUNCTION BLOCKS

Several function blocks performing a single identifiable function may be advantageously grouped together into a 'compound'. This allows easier re-use of the group of function blocks elsewhere in the strategy, and usually clarifies the worksheet layout. The way the function blocks and wiring work is not affected by their being in a compound.

Note: The main (ROOT) FBD is itself a compound.

A compound can be created:

- By creating an empty compound and then cutting and pasting function blocks into it, or
- By creating an empty compound and then dragging and dropping function blocks into it, or
- By first selecting some function blocks and then compounding them.

To cut-&-paste function blocks into an empty compound:

Note: Using cut-&-paste may break some of the function block wiring.

1. Click the Compound toolbutton (or pull down **Make** > **Compound**), then position the Compound cursor on the worksheet and click left to paste it down.
    - The page number allocated to the compound shows in its lower left corner, and the new compound appears in the worksheet 'Contents' list.
2. To move function blocks into the compound, first highlight them, then right-click and select **Cut** from the context menu.
3. Open up the compound by double-clicking it (or double-clicking its name in the **Contents** list). Then right-click in the opened compound window and **Paste** down the function blocks.

Note: To see all opened worksheets at once, select Window > Tile Vertical or Tile Horizontal.

To drag-&-drop function blocks into an empty compound:

Note: Using cut-&-paste may break some of the function block wiring.

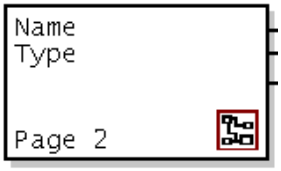
1. Create an empty compound, as described in step 1 of the 'cut-&-paste' method above.
2. Double-click the compound to open it, and position its window adjacent to the window containing the function blocks you want to compound.
    - An easy way to do this is to select **Window** > **Tile Vertical** (or **Horizontal**).
3. Drag the function block(s) to be compounded into the open compound window and release the mouse button. The dragged function blocks add to the compound, and any attached wiring remains intact.

TO COMPOUND SELECTED BLOCKS:

1. Start by highlighting the function blocks you want to compound.
2. In the Edit menu, select Create Compound. The compound appears in the FBD, and its default name appears in the Contents pane.

TO NAME A COMPOUND:

1. Type a name for the compound into the Name field, visible in the Compound tab.
2. The compound Type field can be edited if required.
    - The compound's Name, Type, Page number, and compound icon appear as shown:



TO OPEN AND CLOSE A COMPOUND:

Open a compound to reveal its contents by double-clicking it in the worksheet.

- To see external wiring into and out of the function blocks in the open compound, use the **Editor** tab of the **Settings** dialog.

Close an open compound by clicking the Go Up toolbutton.

TO DELETE A COMPOUND:

Highlight the compound and press the **Delete** button.

ADD COMMENT TAB TEXT

Textual comment attached to a function block, step or transition can be used to describe the purpose and action, or any other information that may be helpful to control strategy developers. It can be configured to appear only when selected or not at all, but the Comment tab Text symbol indicates that there is attached text.

To add Comment tab Text:

1. Double-click the object on the worksheet to see its **Object Properties pane**.
    - If necessary, click on the **Comment** tab to reveal the **Comment text**.
2. Click in the **Comment** tab to locate the text cursor and type in the text.
    - This is a What You See Is What You Get (WYSIWYG) tab.

IMPORT A CONFIGURATION FROM ANOTHER FILE

The user might wish to import a complete Sequential Function Chart (SFC) or Function Block Diagram (FBD) configuration from another file into the current worksheet window, Thus saving unnecessary repetition when the required configuration is similar to an existing one.

To import a configuration from another file:

1. From the editor window **Edit** menu, select **Paste From File…**.
    - Alternatively, right-click anywhere in the current window and select **Paste From File…** from the displayed menu.
2. An **Open** dialog appears, which lists files of the same type as the current file.
3. Navigate to the file you want to import and click **Open**.
    - For a **LIN Database**, the complete FBD layout is pasted down on the current worksheet window.

Note: Function blocks in the imported Database having the same names as function blocks in the current Database are imported, but their names are automatically placed in "quotes" to force you to rename them.

    - For a **LIN Sequence**, the ROOT SFC is pasted down on the current worksheet, and all Actions associated with Steps in the ROOT are also imported. You will see them listed in the Contents pane.

Note: Actions in the imported Sequence having the same names as actions in the current Sequence are not imported. Unassociated Actions are also not imported.

4. The pasted-down layout may obscure part of the current layout. If necessary, **while the imported layout is still highlighted**, drag it to a different position on the worksheet.

20.2.10 Wire the function blocks

Two distinct types of 'wiring' can be created using the Wire toolbutton, depending on which Editor is being used:

1. LIN Database wiring - link function blocks in a LIN Database.
2. LIN Sequence wiring - link the Steps in the Sequence..

Note: LIN Database wiring can also be made during Online Reconfiguration, and is indicated via the change of colour, unless configured otherwise by the User.

CREATE WIRING BETWEEN FUNCTION BLOCKS

Note: You can create and edit wires to and from fields displayed on the Blocks tab and Connections tab in the Object Properties pane using the commands provided on the context menu.

To wire between block fields

Note: To keep network communication traffic to a minimum do NOT wire to cached function blocks. All blocks in the primary control path should remain in the local database wherever possible to avoid complexity of considering failure modes.

1. Locate the cursor over the source Block category symbol, at the lower right corner of the function block. A Database wire symbol appears below the regular arrow cursor.
    - For a compound, locate the cursor over the compound icon instead.
      Do not open the compound!
2. Click the icon to reveal a picklist of available source fields for the connection. Fields with a '⊞' in front contain subfields. View these by clicking the '⊞', or double-clicking the fieldname.

Note: Close the picklist menu by pressing the computers' <Esc> key.

3. Double-click a source field. The menu closes and the cursor displays a single-wire symbol below it.
4. Now locate the cursor over any part of the destination block or compound's outline. A target symbol appears below the cursor and a dashed line 'stretches' from the source to the destination function block.
5. Click over the function block or compound to display a picklist menu of available destination fields. Double-click the required destination field (or subfield) to close the menu and complete the wire connection.

Note: Closed loops. A small red circle appear round a wire arrowhead, indicates that a closed wiring loop has been created. Use the Loopback facility to determine how this loop executes.

6. If required, re-route or 'tidy up' the wire by dragging portions of it to new positions. You can move whole segments of wire, or individual corners (which highlight as small black boxes).

Notes:
1. It may be convenient, when carrying out function block wiring for example, to enlarge a part of the worksheet. This is done by clicking on the function block of interest, to highlight it, then clicking the Zoom in toolbutton at the top of the worksheet to zoom into the selected function block. The Zoom out toolbutton reduces the view magnification.

2. A short 'stub' of wire always remains adjacent to the function block symbol. This may be dragged to different positions but not removed. Wiring can be drawn only vertically and horizontally.

EDIT WIRING BETWEEN BLOCKS

The 'Edit connection' command in the Connections tab context menu is used to change the source (**IN**) or destination (**OUT**) wired to/from the field after confirming that a break in the wire is accepted.

1. Open the Object Properties pane and select the Connections tab to show the fields of the block.
2. Right click in the field that is to be edited to display the context menu.
3. Select the Edit connection command to display a picklist of all the blocks in the database.

Note: Use the filtering icons and drop-down menu on the picklist to ensure only the required data types are available when configuring Variable Line tab parameters.

4. Locate and expand the required block, and select the field to be wired to.

Note: The Undo command and Redo command can be used to reverse the last edit and reverse Undo command respectively.

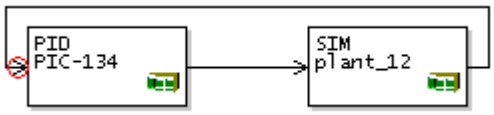
INSPECT WIRING CONNECTIONS

To inspect an existing wire:

1. Locate the cursor over the wire - which turns pink - and double-click it. A **Connections tab** appears at the foot of the worksheet showing the source(s) in the first column and destination(s) in the second column of any wire(s) in the connection.
    - You can use the **Editor** page of the Settings dialog to make a 'tooltip' box appear showing the sources and destinations in a highlighted wire.
2. To force a 'break' in a looped wire, click the **Loopback** box to toggle its TRUE/FALSE value. With **Loopback** TRUE, a black circle appears around the wire's arrowhead, telling the block execution-order algorithm where it is to break the loop, i.e. that this function block input can use data that is one scan old.

WHAT IS A LOOPBACK?

If a wire forms a **closed loop** between function blocks running in the same LIN Database (or assigned to the same user task in a T600 database), one of the wires in the loop acquires a small red circle around its arrowhead, e.g.:



Function blocks in the LIN Database are updated at runtime in an automatically-determined order, designed to ensure that the function block producing data at the source of a connection is updated before the function block that receives this data. When a loop is formed, this rule cannot apply. One of the function blocks in the loop - the first one to be updated at each scan - must use data from the preceding update scan of the user task, i.e. data that is one scan old.

- The function block execution order can be inspected by clicking the Order toolbutton.

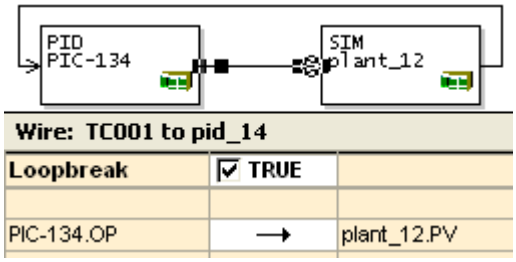
The connection going into this initially-updated function block is the one that is marked in red, to alert you to the situation.

Note: If a wire is added that forms more than one loop, the corresponding number of red circles will appear.

The function block execution order that LINtools determines for a closed loop will often be satisfactory. If, however, the function block shown with a red circled input is not the one which is to be updated first, the loop can be broken at another point (see below) - i.e. its 'beginning' can be redefined.

To break a loop:

1. Click the wire going into the block that is to be the 'start' of the loop. The wire highlights, and its **Connections tab** appears.
2. In the **Connections tab**, click the **Loopback** checkbox to set it to **TRUE**. The selected wire's arrowhead acquires a black circle to show it has been manually selected as the start of the loop, and the red circle disappears.



Note: The black circle indicates to the target instrument that this function block input is permitted to use data that is one scan old, and the function block update ordering will be calculated accordingly. A black circle does not force the data to be delayed, it just tells the function block execution-order algorithm where you prefer it to break the loop. if so desired, more than one wire in a loop can be marked with Loopback status, or wires can be marked that do not at present contribute to loops.

DELETE A WIRE CONNECTION

You can delete an individual connection in a wire, or the whole wire bus.

To delete an individual connection in a wire:

1. Click (or double-click) the wire to display its **Connections tab**.
2. Right-click the connection's entry in the **Connections tab** to display a **Delete Connection** button, then click the button.

To delete the whole wire bus

1. Locate the cursor over the wire - which turns pink - and click it once to highlight it. Wire corners are marked with small black boxes - handles, used to re-route the wire.
2. Press the computer's <**Delete**> key to delete the wire and all connections bussed within.

Note: To restore the deleted connection(s), click the Undo button.

INSPECT A FUNCTION BLOCK EXECUTION ORDER

To inspect a function block execution order:

Click the Order toolbutton to display the function block execution order (if available) as a set of numbered blue boxes attached to the top right corner of each function block symbol. The numbering starts from '1'.

In instruments with several 'tasks', each Task has a separate and independent uniquely-coloured series of execution order numbers, each starting from '1'.

- Grey-coloured numbers denote function blocks that have not yet been allocated to a Task, or that cannot for other reasons show a meaningful execution order, e.g. cached function blocks.

CREATE FBD LAYOUT

To create a graphic function block layout from a block list view:

In the LIN Database Editor, select the **Edit** menu and then **Create FBD layout**. An auto-created Function Block Diagram (FBD) appears. Edit the layout and wiring if required.

20.2.11 Add text to the worksheet

CREATE AND EDIT TEXT ITEMS

This is used to insert and edit text items to the worksheet, but does not follow the same rules as the Comment tab Text.

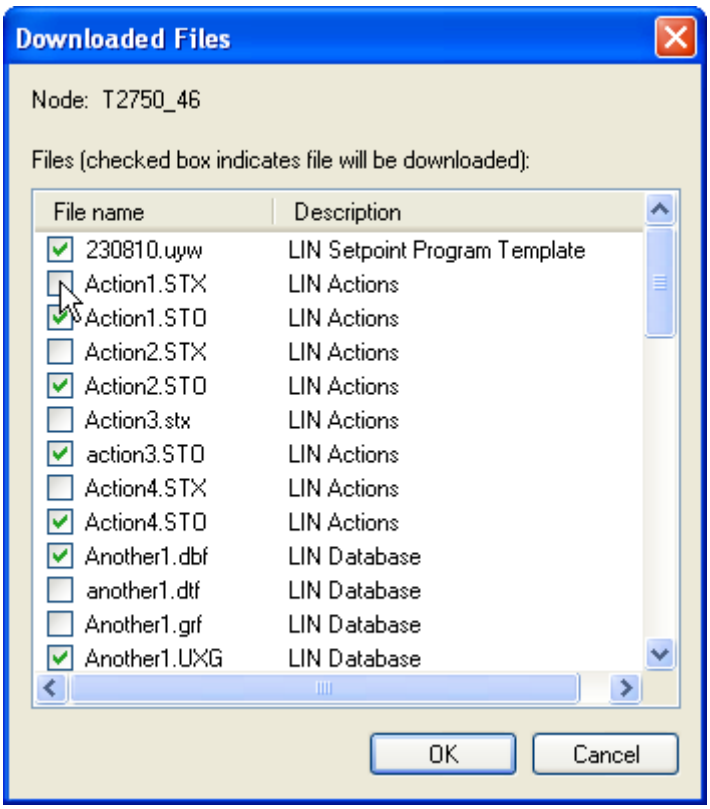
To create textual items on the worksheet:

1. Click the Comment Text toolbutton (or pull down the **Make** menu and select **Comment Text)**. Position the text cursor on the worksheet and click to paste down a default text frame. It shows a '**No Text'** legend, but is in fact empty.
2. Double-click the highlighted text frame to display a free-format **Text** tab in the **Object properties pane**. Click in the **Text** tab to locate the text cursor there and type in the required free-format text, which appears on the worksheet auto-wrapped inside the text frame.
3. You can resize the text frame by highlighting it and dragging any of the handles.
4. You can drag the text frame to a new location on the worksheet.

20.2.12 Save a LIN Database

FILES TO BE DOWNLOADED - EXAMPLE

All ticked files are downloaded to the instrument when a download is initiated.



SAVE A LIN DATABASE

Three files are generated when a LIN Database is saved. They share the same root filename, but have different extensions according to their function:

.dbf = runtime source file

.dtf = comment text file

.grf = graphics layout file

The LIN Database can be saved to disk at any time during its configuration.

Note: When saving a LIN Database these three files are combined as one file and stored in the Instrument folder.

To save a Database to its existing path/filename:

1. To save the LIN Database to its existing path/filename, click the Save toolbutton (or use the **File** > **Save** pulldown).

Note: The LIN Database cannot be saved if function blocks have "NoName". Use the " character in the Find… dialog to locate any "NoName" function blocks.

2. This displays a warning that the LIN Database file has been modified, and requests confirmation to rebuild the Strategy. Modifications include I/O and Tags changes.

   - Yes, will rebuild the Strategy.
   - No cancelS the Save command.

To save a LIN Database to a new path/filename, Select **File** > **Save As...**, and specify the new path/filename in the **Save As** dialog.

LINtools automatic backup on save

20.2.13 Test the Strategy

A simple way to get an idea of how the strategy will work in a Instrument, without having to configure and connect up the I/O Subsystem, is to use a SIM block to simulate plant behaviour. In the slightly modified strategy, a SIM block simulates a process variable and its response to the changing control output.

To simplify these instructions, the I/O blocks can be ignored. The I/O blocks are applicable to a system that includes an I/O Subsystem. However, by locally adding the LOOP_PID block and wiring the SIM block in the Visual Supervisor database as specified, a load can be simulated, and used to produce an output.

To test the strategy,

1. Delete the wire between the AI_UIO 'PV' block and the LOOP_PID block by clicking on the wire to highlight it then operating the <Delete> key.
2. Place a SIM simulation block on the worksheet. (SIM blocks are in the Control function block category.)
3. Wire the **SIM.OP** field into the **LOOP_PID.PV** input, and **LOOP_PID.OP** into **SIM.PV**. This forms a closed loop (as indicated by the small red circle).
4. Name the SIM block, 'sim', and edit both its **NoiseMax** and **Lag1** fields to **25.0**.
5. Save the modified strategy

Notes:

1. The SIM block generates pseudo-random noise at its output *OP*, which simulates a varying process variable input to the LOOP_PID control block's *PV* field. The LOOP_PID block's resulting control output *OP* is fed back to the SIM block's *PV* input, where it is delayed and added to noise to simulate the plant's response to the control input.
2. Increasing the value in the NoiseMax and Lag1 fields will assert Alarms.

20.3 CONFIGURE THE I/O

Note: The I/O Configurator is not available for all instruments. The I/O configuration is not displayed in Monitor mode.

The following stages occasionally overlap, and their order can be varied, especially when editing an existing I/O instrument. Function blocks (including I/O blocks), wiring, and parameter values can be added, modified, or deleted at any time using the LINtools toolkit.

Note: If the I/O Node already exists on the network this process can be avoided by using the Match Hardware command. This command attaches to the target instrument automatically adding blocks that match the hardware fitted exactly.

1. For a new I/O configuration, open the I/O configurator in the **Contents tab** to display the I/O table sub-window.

   Alternatively, with the **Content tab** in tree view, press the '+' until the I/O sub-window is displayed.

Notes:

1. If editing an existing I/O instrument the I/O table cells and Content list I/O slots may already be complete, but can be deleted using the appropriate Delete command from the context-sensitive menu.
2. Pressing the PC's <Ctrl-F6> keys cycles the open sub-windows.

2. In the I/O table sub-window, configure the I/O Node.

   iii. Enter the I/O Node Name in the appropriate cell.
   iv. Double-click the I/O Node Type cell and select the required I/O Node Type from the drop-down list.

3. Then, specify each of the I/O modules required.

   As each of the following steps is completed, Module icons appear in the Contents list. This indicates that an I/O Slot compound containing a MOD_UIO block has been successfully created.

   i. Enter the Module Names in the appropriate slot cells.
   ii. Double-click the corresponding Type cell and select the required Module Type from the drop-down list.
   iii. If the I/O hardware is present on the network, use the Match Hardware command to automatically complete the I/O table.

Note: Existing Module configurations can be moved or copied (using the Ctrl key) by dragging and dropping a required Module in to a different slot.

4. Now, configure the required Module Channels.

Note: I/O Channel blocks do not have to remain in the Compounds in which they were created. Move the I/O Channel block to an appropriate location in the database to simplify the wiring. The compounds are used to simplify the identification of the Modules when first created.

As each of the following steps is completed, '+' icons appear in the **Contents tab**. This indicates that the specified number of Channel blocks have been successfully created in the I/O Slot compound.

i. Double-click a configured Module Slot number to reveal a list of available channels.
ii. Enter the Channel Names in the appropriate channel cells.
   - Right click to display a context-sensitive menu and select Autocreate Channel blocks to create individual Channel blocks or,
   - Right-click the Module Slot number to display a context-sensitive menu and select Autocreate Channel blocks to create all Channel blocks.

Note: To move an existing I/O Channel to another location, change the MOD_UIO blocks SiteNo or Channel field as required.

5. Finally, save the I/O Configuration to disk.

20.3.1 Use the I/O table

This table is used to create the LINtools blocks of the I/O hardware. From this table the following I/O Node blocks can be configured.

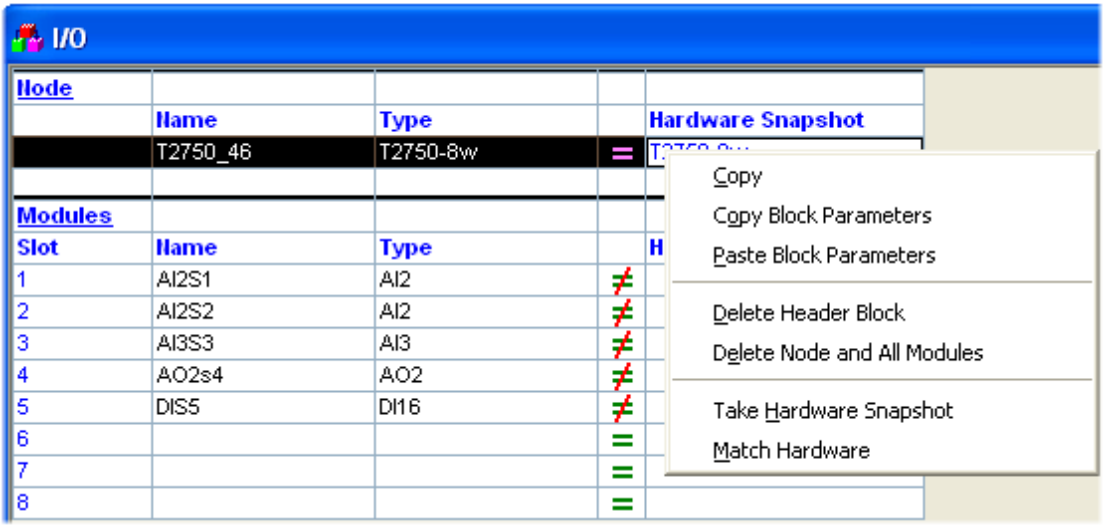
- Modules
- Channels

Note
The I/O Node parameters are automatically entered when the Header block is created.

TO SET UP THE I/O TABLE:

Set up the I/O Node parameters

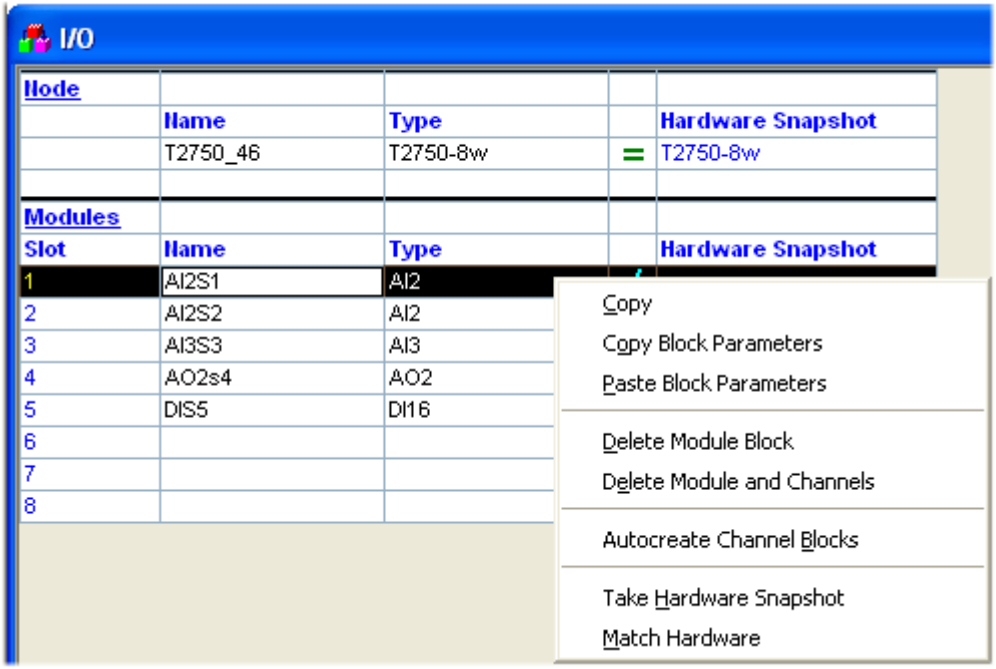
1. Right-click in any cell or column heading, to display a context-sensitive edit menu. E.g. :



2. Select one of the menu items.

Set up the I/O Module parameters

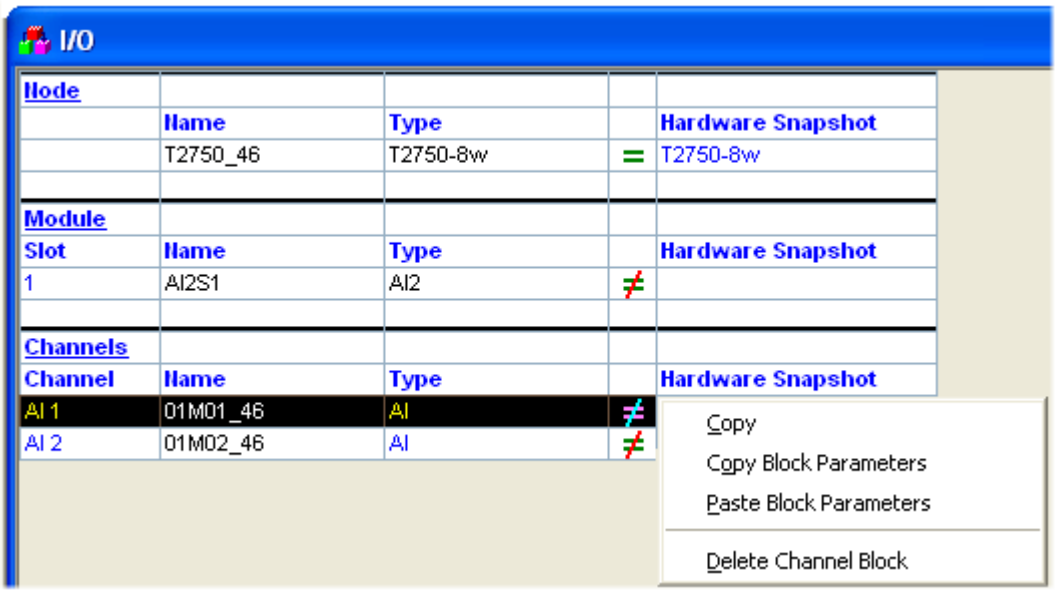
1. Right-click in any cell or column heading, to display a context-sensitive edit menu. E.g. :



2. Select one of the menu items.

Set up the I/O Channel parameters

1. Right-click in any cell or column heading, to display a context-sensitive edit menu. E.g. :



2. Select one of the menu items.

ENTERING DATA INTO THE I/O TABLE

Entering Node data

Enter an appropriate name in the I/O table Name cell. This can be up to 16 characters long.

Entering Module data

1. Next, enter an appropriate name in the Module Name cell.
   - Create the Channel blocks.
2. Select the required Module Type from the drop-down menu in the Module Type cell.

Entering Channel data

1. To display the Channel configuration, simply double-click the required Module slot number. This will display the Channel configuration of the selected Module Type.
2. Create the Channel Blocks in the Channel Name cell. This can be up to 16 characters long.

Note: Inappropriate blocks and nodes can be removed from the database, simply by selecting the required Delete… command.

EDITING DATA IN A I/O TABLE CELL

- Click once on the cell to insert a text cursor at the clicked point. Use this to edit the text.
- Double-click the cell to highlight all its characters. These can now be deleted or overtyped.
- Double-click the cell in the Type column to access the Node or Module types. Click the drop-down icon to reveal available type options.

Note: Additional read only columns are displayed when editing Online I/O. The Name Running in DB column shows the name of the Database currently running in the instrument, the Type in Running DB shows the Type currently running in the instrument. The Hardware Snapshot comparison cells shows the results of comparing a Hardware Snapshot and the I/O Node at this address. If any items differ from the running database, the Online Reconfigure symbol appears in the first column.

20.3.2 Create Modules

ENTER THE MODULE NAMES

This field is used for editing the name of the Module fitted at the selected slot.

To edit the Module name

1. Double-click the Module name cell.
   - Click again to locate the text cursor if editing an existing Module name.
2. Enter maximum 16 characters for the name of the Module.

Note: Using the Module Type and Slot Number (AI4S01) as the Module name may assist when browsing the network.

SELECT THE MODULE TYPE

This field is used to define the type of Module fitted at the selected slot.

1. Double-click the Module Type cell that corresponds to the slot being edited.
2. Click the Picklist icon to reveal available Module type options.
3. Select the Module Type that matches the I/O Instrument hardware.

CREATE CHANNEL BLOCKS

The required Channel blocks can be created either by

- using the **Autocreate Channel Blocks** command or by
- typing the Channel Block name in the appropriate cell.

Note: Using the Slot Number then Channel Number, and Address (e.g. 03M03_00) as the Channel name may assist when browsing the network. This is the naming convention of automatically created Channel names.

To create all the Channel blocks for the Module automatically

1. Right-click the Module Slot number cell to display the context-sensitive menu.
2. Select the **Autocreate Channel Blocks** command. As the command comes to a conclusion the Module icon in the **Contents tab** expands to reveal the Channel Blocks.

To create an individual Channel block for the Module automatically

1. Right-click the Channel Name cell to display the context-sensitive menu.
2. Select the **Autocreate Channel Blocks** command. As the command comes to a conclusion the Module icon in the **Contents tab** expands to reveal the Channel Blocks.

To create an individual Channel block for the Module manually.

1. Double-click the required Channel Name cell.
2. Type in a maximum of 16 characters.

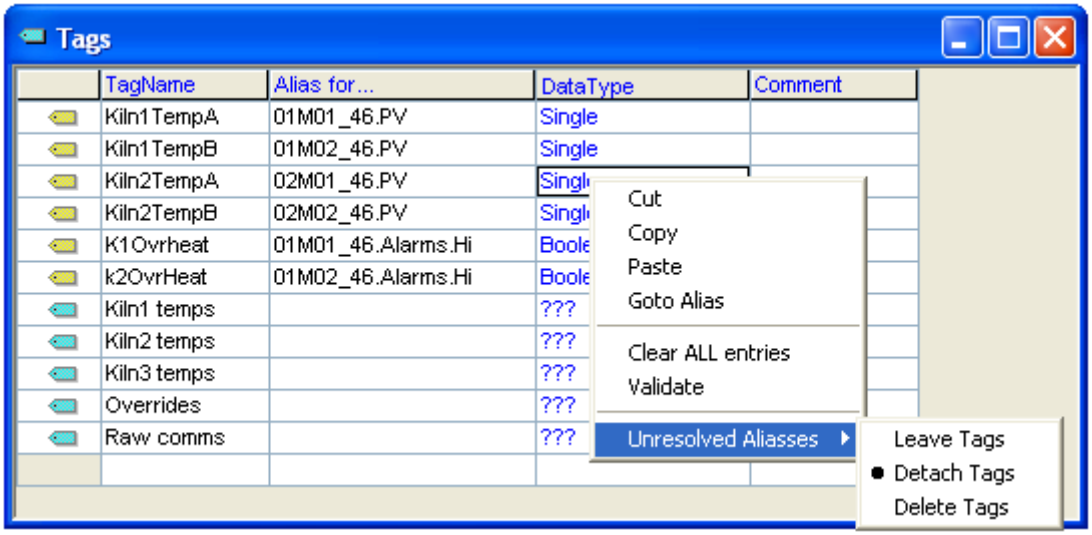
20.4 CONFIGURE TAGS

20.4.1 Use the Tag table

This window is used create Tags and Aliases for Function Blocks and Ladder Programs on the function block instance. A TagName can be created to alias any block, field or subfield.

To set-up or to validate a Tag table:

1. Right-click in any cell or column heading, to display a context-sensitive edit menu, e.g.,



2. Select one of the menu items.

ENTERING DATA INTO THE TAG TABLE

1. Enter an appropriate name in the TagName cells. This can be up to 16 characters long.

Note: When entering TagNames it may be quicker to use the Cut, Copy and Paste commands from the context-sensitive menu.

- All Tags can be removed from the Database file by selecting the Clear ALL Entries command from the context menu.

2. Next, double-click the cell in the Alias for… column to access the Database blocks, etc.. Click the  button to reveal available Database name options.

   Use standard Windows navigation to locate the block, field or subfield.

Note: An Alias can be entered as a placeholder, but this will cause syntax errors until TagName cell is complete. This will then cause a specific name error the Alias For… cell that can be cleared by selecting the required block, field, or subfield.

3. If the Alias is accepted by **LINtools**, the appropriate DataType is automatically entered in this field.

4. If required, enter a brief description for this Tag configuration in the Comment cell.

5. Finally, check this I/O instruments Tag configuration is correct by selecting Validate from the context-sensitive menu.

Note: Any incomplete Tag reference is indicated by a blue Tag icon preceding the TagName.

EDITING DATA IN A TAG TABLE CELL

Double-click the cell to open it, then

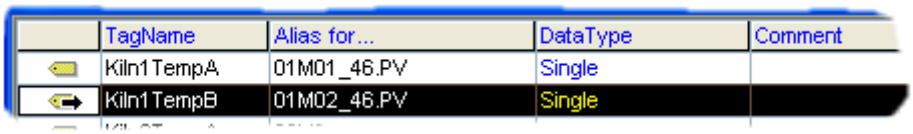
- click once to insert a text cursor at the clicked point. Use this to edit the text.

- double-click the cell to highlight all its characters. These can now be deleted or overtyped.
- double-click in the Alias for… cell to access the Database blocks PickList, etc.. Click the ... button to reveal available Database name options.

COPYING TAG ROWS TO THE CLIPBOARD

You can select one or more rows in the Tag table and copy them to the clipboard.

1. Locate the cursor on the row select button at the left-hand end of the Tag row you want to copy. A small arrow appears over the button - e.g..



2. Click the mouse to highlight the complete row.
   - To select a block of adjacent rows, hold the mouse button down and drag up or down the table to highlight more rows as required. Then release the mouse button.
3. Copy the highlighted row(s) to the clipboard using <**Ctrl+C**>.
   - Alternatively, right-click then select **Copy** from the context-sensitive menu - or use the **Edit** > **Copy** menu.
4. Paste the rows into the Tag table and edit as required.
   - Alternatively, paste as tab-separated text into a suitable text editor or word processor (e.g. Notepad).

Note: Duplicated TagNames pointing to identical blocks, fields, or subfields cause a error, indicated as red underline Tag Name text.

20.4.2 Create Tags and Aliasses

To enter a Tag Name in the Tag table:

1. Double-click the first available **TagName** cell.

Note: Generating Block Tags is not permitted.

2. Type in an appropriate name for the block, field or subfield about to be aliased.

   By using the Settings dialog a TagName of upto either 8 or 16 characters can be configured. A TagName that exceeds the usual 8 characters for a function block is indicated by a Block Tag (yellow Tag icon including a slightly obscured square), and will inhibit (read-only) the **Alias for** field.

Note: Any incomplete Tag reference is indicated by a blue Tag icon in the column preceding the TagName. A successful Tag reference to a function block field/subfield is indicated by a yellow Tag icon.

Select the Alias For...

To select what the TagName is an alias for:

1. Double-click the cell in the Alias for… column to access the LIN Database blocks, etc..

Note: Editing the alias for a Block Tag is not permitted. This is indicated by the colour change of the Block Tags 'Alias For…' field.

2. Click the button to reveal available LIN Database name options.

   Use standard Windows navigation to locate the block, field, or subfield.

Note: An Alias can also be typed into the cell as a placeholder.

Edit the DataType

This **DataType** field shows the type of data recorded at the location indicated in the **Alias for...** field. It cannot be edited, however it does change if the **Alias for...** field is changed.

Enter Comments

Comments are used to give information which should be helpful to control strategy developers.

To select what the Tag Name is an alias for:

1. Double-click the appropriate Comments cell to access the Database blocks, etc..
2. Type in any appropriate information as required.

20.5 CONFIGURE SEQUENCES

20.5.1 Create a LIN Sequence file

The best place to create a LIN Sequence file is via the **Contents** Pane or using the **File** > **Get me started** command, but it can also be created using the New LINtools file toolbutton or in the Project environment. That is, in the Explorer view of the Project file, inside the instrument folder that will run the LIN Database.

Note: A new Sequence file can be created elsewhere - e.g. on the desktop - but the file will need additional configuration to tell it the instrument type, etc.

TO CREATE THE NEW LIN SEQUENCE FILE:

1. To create a Specific Sequence file, select **New** > **LIN Sequence**. For a Generic Sequence, select **New** > **LIN Generic Sequence**. A file is created with a default name. The filename extension is **.sdb** for a Specific Sequence, or **.uqd** for a Generic Sequence.
   - Alternatively, select **File** > **Get me started** and select the create new LINtools file radio button. Context related information that is displayed in the dynamic help window can be used to help decide the file type required.
2. Rename the LIN Sequence file if required.

20.5.2 Access the LIN Sequence Editor

Note: The 'Get me started' wizard can assist when opening files that have recently been edited.

TO ACCESS THE SEQUENCE EDITOR:

1. Double-click the sequence file or its icon to launch the **Sequence Editor**.
   - The Sequence Editor can also be launched be selecting the 'Open Sequence file (.sdb)' command from the context menu in the LIN Database, accessed from the .

Note: If the file specified in the FileName field of the LIN Database does not exist, it is automatically created as the Editor opens.

2. If the Sequence is already opened, the default DBF will automatically be chosen and LINtools will load the LIN Database that was last used with this Sequence. If a LIN Database is not associated with the Sequence the **Load Database for Sequence Editing session** dialog is displayed. Use this to open the LIN Database that the Sequence is to target. Select the required **.dbf** file and **Open** it.

Note: If you don't want to open a target database right now, click the dialog's Cancel button. You can always open one later, which will make configuring specific Actions and Transitions much easier.

If accessing a **Generic Sequence**, you may be told '**No map file loaded**' and asked if you want to load one.

- Click No if you are creating a new Generic Sequence, or if no relevant map exists. A new blank Map file will be created, that can then be edited and saved.
- Click Yes if a map exists that you want to use, e.g. for creating a Specific Sequence from a Generic Sequence. Locate and open the map using the **Load Map for Sequence Editing session** dialog that appears.

3. The LIN Sequence Editor window now appears, displaying the **Contents** pane (empty for a new Sequence). The Contents tab at the left of the window lists this worksheet as '**ROOT (Chart)**' (FBD in List View), I/O (I/O configuration including appropriate slots), and Tags.

Note
'Chart' refers to the Sequential Function Chart (SFC).

Customise the default colour coding

20.5.3 Associate a Sequence with a Database

To make a Sequence operate in conjunction with a Database a special function block (SFC_CON) must be included in order to associate the Database with the Sequence.

The SFC_CON block (in the BATCH function block category) includes two fields (FileName and Filepath) that specify the Sequence to be loaded and run by the Database. Each Sequence to be run must have its own SFC_CON block configured in the Database.

In multi-tasking instruments, each SFC_CON block (the interface between the Database and the Sequence) and any blocks interfacing with a Sequence must always operate on User Task 4.

Note: The FileName field shows a 'page' icon ( ) indicating a different file must be edited to ensure the correct operation of the Strategy.

20.5.4 Use the Find utility

In a large configuration it can be difficult to find a particular function block, text string, action, step, etc. This utility helps to locate such objects, as follows:

1. Click the 'Find...' item in the Edit menu. The 'Find...' display appears, containing one or more tabs, depending on the context.
2. Click a tab to display the required page:

- a Block page - finds function blocks.
- a Field page - finds function block fields.
- a Text page - finds text strings in Actions, comments, and ladder diagram fields.
- an Action page - finds chart, ladder, and text Actions.
- an Association page - finds Action associations.
- a Step page - finds Sequence Steps.
- a Tag page - finds Tags.
- a Changes page - finds changes between the Strategy files on the Computer and the 'live' Instrument.

Note: Some pages contain buttons that are identical to the commands available on the context menu.

20.5.5 Configure steps

PLACE A STEP

1. To place a Step, click the Step toolbutton at the top of the window, or click on the 'Step' item in the 'Make' menu to 'load' the cursor with a Step symbol.
2. Click the cursor on the worksheet to paste down a default (auto-numbered) Step. The highlighted Step is marked with a Step wire symbol in its lower right corner.

Notes:

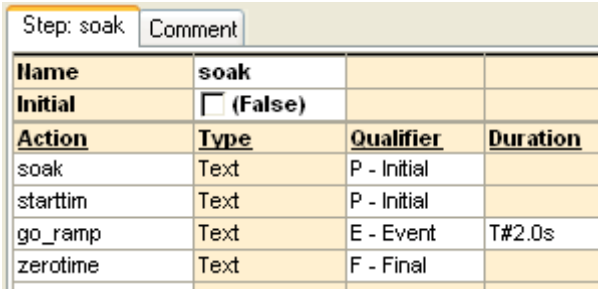
1. An invisible grid helps align the pasted Steps.

2. To place a series of steps via the loaded cursor, hold down the computer's <Ctrl> key as the step is pasted. This leaves the cursor still loaded and ready to paste more steps.
To delete a step, click on it once to highlight it, then right-click and select Delete in the context menu. To delete several steps, multiple-select them, then Delete as above.

CONFIGURE A STEP NAME & INITIAL STATE

To configure the name of a step and its initial state:

1. Double-click anywhere within the step symbol (except over the 'wire' symbol) to open the **Step tab** at the foot of the worksheet. The white areas are editable.
2. Click the white Name field and type in a name for the step. Hit <Return> to enter the new name.
3. Click the Initial checkbox to toggle the setting between **TRUE** and **FALSE**. 'TRUE' makes the step an initial step, i.e. one that becomes active when the Sequence starts.



ASSOCIATE AN ACTION WITH A STEP

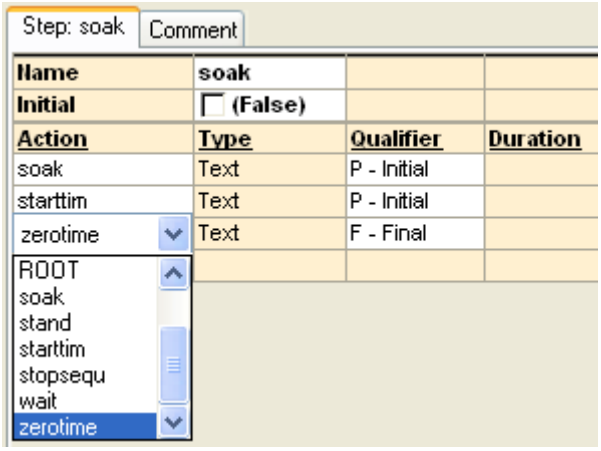
When a Sequence step becomes active, each Action associated with the Step is executed in the order listed in the **Step** tab on the **Object properties pane**, and with a timing specified by the action's qualifier.

To associate an Action with a Step:

1. Double-click the step to see the **Step tab**, normally at the foot of the worksheet. Select the **Step tab** if necessary.

   An empty white box appears at the foot of the **Action** column in the **Step tab**.

   - If this is the first Action you are associating with this Step, the white box is the only item seen in the table.
2. Double-click the white box and click on an existing Action from the pulldown menu. The selected Action is added to the table with a default **Qualifier**. The **Type** entry automatically shows the type of Action selected (**Text**, **Ladder**, or **Chart**) and is read-only.

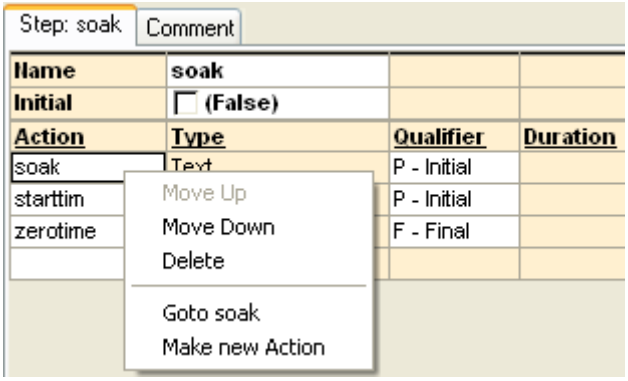


3. Edit the default Action Qualifier entry if required, by clicking it and selecting from the dropdown menu. Hit <**Return**> to enter your choice.
4. For qualifiers needing a time specification, a default entry appears in the **Duration** column with the format **T#...**. Edit the time value by double-clicking the entry to display the Step Time Setup dialog.
5. Type in as required the **Days**, **Hours**, **Minutes**, and **Seconds** values, and click **OK** to enter them. You can type valid integer and/or decimal numbers into the fields; they are automatically rationalised and displayed in the most economic format when entered.
6. To associate further Actions with the step, repeat steps **3** to **6** as required.

Note: The associated Actions are initiated in the order displayed in the Step tab, starting at the top-most Action. The execution timing of each Action also depends on its action qualifier.

7. To alter the execution order of the Actions, remove an association from the Step, or open up an Action window, right-click on the Action's row to display a context menu, then click the required item:

Note: All Action Associations can be hidden or shown by pressing the Action Associations toolbutton or View > Show Action Associations.



ACTION QUALIFIERS

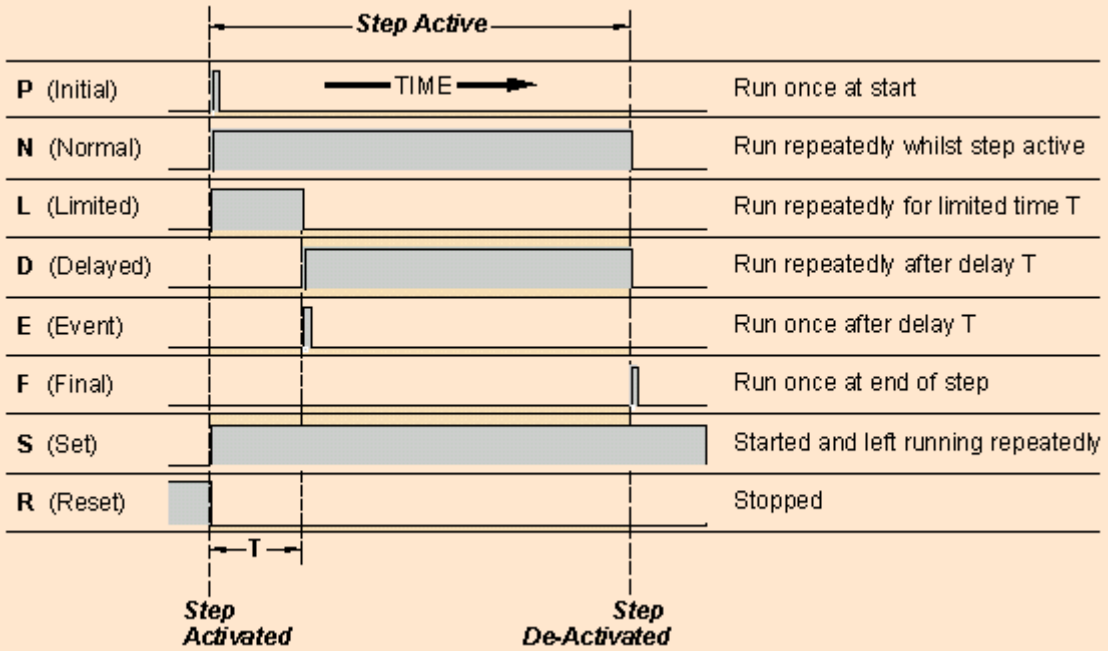
When an Action is associated with a LIN Sequence Step, a suitable **Action Qualifier** must be selected to specify how the Action will be timed when the Step activates.

- A LIN Action can also be associated with its controlling Action-type function block via an Action Qualifier.

The possible action qualifiers are:

- **P (Initial) -** The Action runs once only, at the start of the step's activity.
- **N (Normal) -** The Action runs repeatedly for as long as the step is active.

- **L (Limited) -** The Action runs repeatedly for a limited time from the start of the step's activity.
- **D (Delayed) -** The Action runs repeatedly after a delay until the end of the step's activity.
- **E (Event) -** The Action runs once only after a delay.
- **F (Final) -** The Action runs once only, at the end of the step's activity.
- **S (Set) -** The Action starts running at the start of the step's activity, and is left running repeatedly.
- **R (Reset) -** The Action stops running at the start of the step's activity, and is left halted.



ADD COMMENT TAB TEXT

Textual comment attached to a function block, step or transition can be used to describe the purpose and action, or any other information that may be helpful to control strategy developers. It can be configured to appear only when selected or not at all, but the Comment tab Text symbol indicates that there is attached text.

To add Comment tab Text:

1. Double-click the object on the worksheet to see its **Object Properties pane**.
   - If necessary, click on the **Comment** tab to reveal the **Comment text**.
2. Click in the **Comment** tab to locate the text cursor and type in the text.
   - This is a What You See Is What You Get (WYSIWYG) tab.

IMPORT A CONFIGURATION FROM ANOTHER FILE

The user might wish to import a complete Sequential Function Chart (SFC) or Function Block Diagram (FBD) configuration from another file into the current worksheet window, Thus saving unnecessary repetition when the required configuration is similar to an existing one.

To import a configuration from another file:

1. From the editor window **Edit** menu, select **Paste From File…**.
   - Alternatively, right-click anywhere in the current window and select **Paste From File…** from the displayed menu.
2. An **Open** dialog appears, which lists files of the same type as the current file.
3. Navigate to the file you want to import and click **Open**.

- For a **LIN Database**, the complete FBD layout is pasted down on the current worksheet window.

Note: Function blocks in the imported Database having the same names as function blocks in the current Database are imported, but their names are automatically placed in "quotes" to force you to rename them.

- For a **LIN Sequence**, the ROOT SFC is pasted down on the current worksheet, and all Actions associated with Steps in the ROOT are also imported. You will see them listed in the Contents pane.

Note: Actions in the imported Sequence having the same names as actions in the current Sequence are not imported. Unassociated Actions are also not imported.

4. The pasted-down layout may obscure part of the current layout. If necessary, **while the imported layout is still highlighted**, drag it to a different position on the worksheet.

20.5.6 Sequence wiring operations

TYPES OF SEQUENCE WIRING

There are three ways in which wires can be configured:

- Step-to-Step wiring
- Step-to-Transition wiring - to create a convergence
- Transition-to-Step wiring - to create a divergence

STEP-TO-STEP WIRING

This may be used to create a branch.

To draw a wire from a Step to another:

1. Locate the cursor over the source step's Step wire symbol. A wiring symbol appears below the normal arrow cursor.
2. Click the symbol then locate the cursor over any part of the destination. A Target symbol appears below the cursor.
3. Click over the step to complete the wire connection. A brown (pre-configured) Transition bar appears over the new wire. The highlighted Transition bar can be dragged to a different position if required.

Note: Although Transition bars are created automatically when wires are drawn from one step to another, an isolated Transition bar can be placed in the SFC using the Transition toolbutton.

4. If required, edit, re-route, or 'tidy up' the wire.

Note: A short ‘stub’ of wire always remains adjacent to the Step symbol. This may be moved to different positions but not removed. Wiring can only be drawn vertically and horizontally.

STEP-TO-TRANSITION WIRING

This may be used to create a convergence.

To draw a wire from a Step to a Transition:

1. Locate the cursor over the source step's Step wire symbol. A Target symbol appears below the cursor.
2. Click the symbol then locate the cursor over the destination Transition bar. A Target symbol appears below the cursor.
3. Click the Transition bar to complete the wire. A black double-line appears above the Transition to show convergence. The highlighted Transition bar can be dragged to a different position on the double-line if required.

4. If required, edit, re-route, or 'tidy up' the wire.

Note: Although Transition bars are created automatically when wires are drawn from one step to another, an isolated Transition bar can be placed in the SFC using the Transition toolbutton.

TRANSITION-TO-STEP WIRING

This may be used to create a divergence.

To draw a wire from a Transition to a Step:

1. Click the wire toolbutton (or select Make > Wire) and locate the cursor over the source transition bar. A Target symbol appears below the cursor.
2. Click the transition bar then locate the cursor over the destination step. A Target symbol appears below the cursor.
3. Click the Step to complete the wire. A black double-line appears below the Transition to show divergence. The highlighted Transition bar can be dragged to a different position on the double-line if required.

Note: Although Transition bars are created automatically when wires are drawn from one step to another, an isolated Transition bar can be placed in the SFC using the Transition toolbutton.

EDIT SEQUENCE WIRING

To edit wiring layout

Re-route or 'tidy up' the wire by dragging portions of it to new positions. Whole segments of wire, or individual corners (which highlight as small black boxes) may be moved.

Note: A short 'stub' of wire always remains adjacent to the Step symbol. This may be dragged to different positions but not removed. Wiring can only be drawn vertically and horizontally.

To delete a wire segment

1. Click the relevant wire segment to highlight it.
2. Right-click and select 'Delete' from the context menu that appears.

To delete a Transition and associated wires

1. Click the Transition bar to highlight it.
2. Right-click and select 'Delete' from the context menu that appears. The Transition and its wire(s) are deleted.

20.5.7 Configure Transitions

CREATE A TRANSITION EXPRESSION

1. In the SFC worksheet, double-click on a **Transition bar** to display a **Transition tab** in the **Object properties pane**, normally at the foot of the worksheet. The **Transition tab** is titled with the source and destination steps linked by the Transition - e.g. 'STEP1 to STEP2'.

Note: Pre-configured Transitions have brown bar symbols. When configured the bars appear black.

2. Type in the Structured Text expression for the Transition. Right-click in the pane to see a context menu of items to help you write valid Transition text.
   - Alternatively, to create the Transition as a Ladder Diagram, select **Convert to Ladder** from the context menu. A single-rung ladder diagram appears in the main pane, with an output coil linked to the notional field 'STEP<m> to STEP<n>'. Configure this rung with the required elements to yield a TRUE/FALSE result.

Note: Coil Variable and Type values cannot be edited.

3. To check the validity of the Transition expression you can compile the Structured Text in the Transition.

   - Alternatively this can be done automatically when the LIN Sequence is saved.

Note: All Transition Text can be hidden or shown on the Worksheet by pressing the Transition Text toolbutton or View > Show Transition Text.

 Compiling Structured Text

ADD COMMENT TAB TEXT

Textual comment attached to a function block, step or transition can be used to describe the purpose and action, or any other information that may be helpful to control strategy developers. It can be configured to appear only when selected or not at all, but the Comment tab Text symbol indicates that there is attached text.

To add Comment tab Text:

1. Double-click the object on the worksheet to see its **Object Properties pane**.

   - If necessary, click on the **Comment** tab to reveal the **Comment text**.

2. Click in the **Comment** tab to locate the text cursor and type in the text.

   - This is a What You See Is What You Get (WYSIWYG) tab.

20.5.8 Create a Sequence Action

TO CREATE A NEW LIN SEQUENCE ACTION:

1. Pull down the **Make** menu and select **Action…** (or click the Make Action toolbutton). A Make Action dialog appears.

2. Type in a Name for the action and click the **Chart**, **Structured Text**, or **Ladder Diagram** radio button according to the type of Action required. Click **OK** to close the dialog and open a new **Action window**. The Action now appears in the Contents tab with its characteristic icon (in Tree view) or its page number (in Index view).

3. For a **Text action**, type in the required Structured Text statements. For a **Chart action**, configure as for the ROOT (main) chart. For a **Ladder diagram action**, add rungs and ladder elements to specify the required Action.

   - For a Generic Sequence, Actions contain Generic names configured in the Map table Generic names cells instead of **specific** Database function blocks and fields.

Note: Pressing the PC's <F9> key displays the Variable Picklist. A filtered Picklist may be expanded to show all available variables by pressing the <F8>.

4. Check the validity of the Sequence Action by running the Compile function.

TO EDIT A LIN SEQUENCE ACTION:

1. Double-click the LIN Sequence Action in the Contents tab to display its Action window.

2. Edit the LIN Sequence Action by typing and entering Structured Text in the Action window (Text actions), or by editing the SFC (Chart actions), or the Ladder Diagram (Ladder actions).

ADD COMMENT TAB TEXT

Textual comment attached to a function block, step or transition can be used to describe the purpose and action, or any other information that may be helpful to control strategy developers. It can be configured to appear only when selected or not at all, but the Comment tab Text symbol indicates that there is attached text.

To add Comment tab Text:

1. Double-click the object on the worksheet to see its **Object Properties pane**.

- If necessary, click on the **Comment** tab to reveal the **Comment text**.

2. Click in the **Comment** tab to locate the text cursor and type in the text.

- This is a What You See Is What You Get (WYSIWYG) tab.

COMPILE AN ACTION OR TRANSITION

Compiling an Action or Sequence Transition converts it to 'object code'. If LINtools encounters any errors during an attempted compilation it reports them in a 'compiling' pane. Compiling is, therefore, a good way to check the validity of Structured Text or ladder diagrams.

To compile a single Action or Transition:

1. Right-click in the Action window or transition window and select the **Compile [...]** item from the menu. A new pane appears reporting the results of the compiling operation. Any errors found are listed.

- Alternatively, in the **Edit** menu, click **Compile [...]**.

Note: Compiling does not save the object code - only File > Save does. But if you do save the Sequence or actions file you will see a compile report for all the Actions in the file.

2. Double-click an error line in the report to highlight in the action window or transition window the portion of structured text or ladder diagram that needs correcting. (The relevant Action window or Transition window appears if not already on view.)

- Sometimes in a ladder diagram an error relating to the **whole** rung (e.g. 'Bad label') highlights the **end** of the rung. Similarly, an error due to a fault in the **body** of the diagram (e.g. 'Unresolved jump') is only highlighted at the **last item** in the diagram.

3. When all errors have been corrected, recompiling the Action or Transition produces a **'No Errors'** report.

Note: If a generic sequence is being configured, but the map is incomplete, any 'Can't find this name' errors can be ignored, as these errors will be corrected when a specific sequence is generated. Other structured text errors should be corrected now.

20.5.9 Convert sequences

CREATE A SPECIFIC FROM A GENERIC SEQUENCE

A Specific Sequence is created from a Generic Sequence by converting the generic database block and field names into their specific equivalents.

A map is used to determine the specific block/field that each generic block/field will convert to.

To create a Specific Sequence from a Generic Sequence:

1. Open the LIN Sequence worksheet for the existing Generic Sequence to be converted.

2. Access the Map pane.

- In order to start with a completely new empty map, select **File** > **New Generic Map...**. An **Error** message appears asking if extra names are to be added to the map. Click **Yes** to insert all the generic names in the Sequence into the **Generic** column of the map.
- If an existing map is to be used, select **File** > **Open Generic Map...**, then locate and open a map. If the chosen map contains any syntax errors, a dialog tells you this. Otherwise, Validate the loaded map. An **Error** message may appear asking if extra names are to be added to the map. Click on **Yes** to insert all the missing generic names in the Sequence into the **Generic** column of the map.

3. Fill in any blank specific names in the map table's **Field** column.

Validate the map and correct any errors detected. Repeat this until all map errors have been removed, and the '**Map validated OK**' message appears.

Note: The validation process often helps by highlighting a map table cell with a syntax error.

4. Select **File** > **Build Specific SFC File** to display a **Build Specific SFC File** dialog. Type in a root **.sdb** filename in the **File name** box and click **Save**.

5. The new Specific Sequence is compiled and saved as three files. Correct any errors the compile operation reveals.

CREATE A GENERIC FROM A SPECIFIC SEQUENCE

A Generic Sequence is created from a Specific Sequence by converting the specific database block and field names into their generic equivalents.

A map is used to determine the generic block/field that each specific block/field will convert to.

To create a Generic Sequence from a Specific Sequence:

1. Open the LIN Sequence worksheet for the existing Specific Sequence which is to be converted.

2. Access the Map pane.

   - To start with a completely new empty map, select **File** > **New Generic Map...**. An **Error** dialogue appears asking if extra names are to be added to the map. Click **Yes** to insert all the specific fields in the Sequence into the Field column of the map.
   - If an existing map is to be used, select **File** > **Open Generic Map...**, then locate and open a map. If the chosen map contains any syntax errors, a dialogue message appears. Otherwise, Validate the loaded map. An **Error** dialogue may appear asking if extra names are to be added to the map. Click on **Yes** to insert all the missing specific fields in the Sequence into the Field column of the map

3. Fill in any blank generic names in the map table's **Generic** column.

4. Validate the map and correct any errors detected. Repeat this until all map errors have been removed, and the '**Map validated OK**' message appears.

Note: The validation process often helps by highlighting a map table cell with a syntax error.

5. Select **File** > **Build Generic SFC File** to display a **Build Generic SFC File** dialogue. Type in a root **.uqd** filename in the **File name** box and click **Save**.

6. The new Generic Sequence is compiled and saved as three files. Correct any errors the compile operation reveals.

20.5.10 Save a LIN Sequence

Three files are generated when saving a Sequence. They share the same root filename, but have different extensions according to their function:

Specific Sequence filename	Generic Sequence filename	File Type
<filename>**.sdb**	<filename>**.uqd**	Runtime & Structure Text source file.
<filename>**.sdt**	<filename>**.uqt**	Comment text file
<filename>**.sgx**	<filename>**.uqg**	Graphics (SFC) layout file

The Sequence can be saved to disk at any time during its configuration.

To save a LIN Sequence:

1. To save the LIN Sequence to its existing path/filename, click the Save toolbutton (or use the **File** > **Save** pulldown).

   To save the LIN Sequence to a new path/filename, select **File** > **Save As**….

- If you have edited the current map, you are asked '**Do you want to save the current Map?**' Click **Yes**, and specify a map filename in the **Save Map File** dialog.

Note: If you are saving a Generic Sequence, the Map file has a .uqm extension. It is manipulated using the File > Open, Save, New or Validate Generic Map commands.

2. The LIN Sequence is automatically compiled as well as saved, and a compiling report is displayed in a separate pane.

3. You can trace and correct any errors using the compiling report displayed in the Compile Results pane.

Note: If you are saving a Generic Sequence and your map is not completed, you can ignore any 'Can't find this name' errors. You will correct these when you come to make a Specific Sequence from this Generic Sequence. But you should try to correct any Structured Text syntax errors now.

Compiling a Sequence

Automatic backup on save

Saving a LIN Database

20.6 MONITOR REMOTE SEQUENCES

20.6.1 Access the Sequence Action Monitor

You can access the Sequence Monitor in several ways, including direct access from the Database file icon associated with the Sequence or when already viewing an Instrument database via Online connect.

Note: If you access the monitor within a Project instrument file, you will be able to skip several of the following steps. This is because LINtools will already know a lot about the Database, and have all the necessary files to hand.

To access the Sequence Action Monitor from the associated Database icon located in the 'Contents' pane:

1. Right-click the icon of the Database file associated with the Sequence, and select **SFC Monitor** from the context menu.

   - If LINtools does not know the server port communicating with the LIN Sequence, a Port Select dialog appears, enabling you to select a server and port to access the LIN Sequence to be monitored.
   - If LINtools is not already attached to the LIN OPC server, a '**Making attachment to LIN OPC Server**...' message appears.
   - If LINtools does not know the node address of the remote instrument, a Node Address dialog appears, in which you specify the node of the instrument running the Database associated with the Sequence.

2. Choose a Sequence to Monitor dialog appears. Use this dialog to select the Sequence you want to monitor.

   If LINtools cannot find the associated Sequence file (.sdb), a message tells you that the Sequence file selected could not be loaded. In this case, hit **OK** to clear the message. An **Open** dialog then appears, letting you locate the missing .sdb file. Do this, and click **Open** in the dialog.

3. The Sequence Action Monitor window opens, with the root SFC sub-window on display.

To access the Sequence Action Monitor from the associated Database:

This is the simplest method if you are already Online Connected to the Instrument Database.

1. In the Database Monitor window, pull down the **Online** menu and select **View Sequence…**. The **Choose a Sequence to Monitor** dialog appears. Click on the SFC_CON block controlling the Sequence to be monitored, and hit **OK** to open up the selected Sequence for monitoring.

2. To open the Sequence Monitor in a new window, tick the **New window** checkbox before you hit **OK**. You may be asked to select the relevant **LINtools port** and also the instrument's **node address** - see step 1 above.

20.6.2 Sequence monitoring operations

In the Sequence Monitor's chart (SFC) window, the user can operate on individual components in the remote Sequence as follows:

See the currently-active step(s)
The currently-active Sequence Steps are highlighted with a green diamond symbol.

Hold Steps in the active state
Right-clicking on a Step reveals two states:

- **Hold** - 'Holds' the corresponding step in the remote Sequence. When a 'held' step becomes active it stays active indefinitely - even if the following Transition becomes TRUE - and so holds up the Sequence. A held step displays a held symbol.
- **Normal** - Returns a remote 'held' step to normal.

Force Transitions to be TRUE or FALSE
Right-clicking on a Transition symbol reveals three menu items:

- **Hold** - Forces the remote Transition expression to behave as if it were FALSE (even if TRUE), and so blocks the remote Sequence from moving on to the next step(s). The Transition symbol turns **red** to show it is on 'hold' -  .
- **Continue** - Forces the remote Transition expression to behave as if it were TRUE (even if FALSE), and so forces the remote Sequence to move immediately on to the next step(s). The Transition bar symbol changes to a **green** arrowhead to show it is on 'continue' - 
- **Normal** - Returns a remote 'hold' or 'continue' Transition to normal. 'Normal' Transition symbols are **black**.

20.6.3 Sequence Control operations

From the Sequence Monitor window or while online to the instrument, you can control a remotely-running Sequence.

To reset, stop, hold, or run the remote Sequence:

In the Sequence Monitor window's **Online** menu, select **Sequence**. A menu of four items appears:

- **Reset** - Stops all step activity and initialises the remote Sequence, i.e. all actions halt, and step timers are stopped and zeroed. Highlighted steps on the display de-highlight. The associated remote SFC_CON function block's Run parameter sets to FALSE, and Init sets TRUE.

Note: Values written to the database by the Sequence are not un-written.

- **Stop** - Stops all remote step activity, i.e. all actions halt, and step timers are frozen (stopped but not zeroed). Highlighted steps on the display remain so. (The remote SFC_CON function block's Run and Init parameters both set FALSE.)
- **Hold** - Disables all Transitions in the entire remote Sequence, i.e. they act as if FALSE, so blocking any progress. Active steps remain so indefinitely, i.e. step timers and actions continue running, and timed qualifiers remain effective. (The remote SFC_CON block's Hold and Run parameters set TRUE.)
- **Run** - Restarts the remote Sequence from where it stopped, which is from the beginning if it was reset. (Run resets to TRUE).

20.6.4 Monitor a Ladder Diagram Sequence Action

The Sequence Action Monitor allows the user to inspect and monitor Ladder Diagram Sequence Actions as follows:

1. In the LIN Sequence Action Monitor, locate the required action in the Contents pane and double-click it. The Ladder Diagram representing the selected Action opens as a window in the main Monitor display.
   - Alternatively, use the Find utility to locate the required Action and open its window.
2. In the Ladder Diagram, (double-)click any element to open its Element tab, normally at the foot of the window. This shows the element's parameters - **Rung Label**, **Variable**, **Type** - as read-only values.
3. Hover the cursor over or near an element in the Ladder Diagram to reveal a yellow 'tooltip' box indicating the attached Database point and its current value - e.g. **pulse.In = FALSE**.
4. Look at the colours of the different contact and coil elements in the Ladder Diagram. These indicate the current TRUE/FALSE state of the associated variable.

Customise the default colour coding

20.6.5 Choose a Sequence to Monitor

Use this dialog to select a LIN Sequence to monitor in the Sequence Monitor window.

To access the dialog:

In the Monitor window, in the **Online** pulldown menu select **View Sequence…** to display the **Choose a Sequence to Monitor** dialog.

USE THE DIALOGUE

The dialogue lists all the SFC_CON blocks by name - and their corresponding Sequence filenames.

Highlight the SFC_CON block associated with the LIN Sequence to be monitored and click on '**OK**'.

Note: SFC_CON block names listed in the dialog are only those present in the Monitor Database. If there are no Sequences running on the selected node, a "No Sequences Loaded" error message appears.

20.7 CONFIGURE LIN ACTIONS

20.7.1 Create a LIN Action file

The best place to create an Action file is within the **Contents** Pane, but it can also be created using the New LINtools file toolbutton or in the Project environment. That is, in the Explorer view of the Project file, inside the instrument folder that will run the LIN Database.

New Actions file can also be created elsewhere - e.g. on the desktop - but a Template Library will have to be selected every time it is opened, and the actions will not be associated with their target instrument.

Note: This first Action has been automatically made for use with an ACTION block template. If this Action file is to contain methods for blocks from another template library, delete the Action from Contents pane, change the instrument library in the template palette and Make new actions for templates chosen from the selected template palette.

To create the LIN Action file:

1. Select the either Folder displayed in the Contents pane and right-click to reveal a context menu.
   - Alternatively, use the New LINtools file toolbutton to run the New LINtools file wizard.
2. Select **New** and the LIN Action required. An **Add new file to configuration** dialog appears.

3. Enter a concise filename. The extension is added automatically as defined by the selected File type.
4. This new Action file is created as a Structured Text window.
   - Aany further LIN Actions can now be created as either Structured Text or Ladder Diagrams using the ACTION or DIGACT blocks.
5. Edit the Actions file as required.

Note: If the strategy does not require an Action associated with the ACTION block template, remember to delete the Action file before creating a new Action for the templates chosen from the defined template palette.

20.7.2 Access the LIN Action Editor

To access the LIN Action Editor:

1. Double-click the Action file or its icon to launch the LINtools Engineering Studio with all **Action Editor** functions available.
2. The Action Editor window now appears, displaying the **Contents** pane, including a Structured Text file ('**untitled (ACTION)**' for a new Action) and all previously created ACTION or DIGACT configurations.

 Customise the default colour coding

20.7.3 Make a LIN Action

To make a new LIN Action using the Action… item:

Note: If the Function Block Template Palette is open, start at step 3.

1. Right-click in the Action Editor's **Contents** pane - or click the **Make** icon - to display a menu.
2. Click **Action…**. The **Function Block Template Palette** opens, listing available action function block types.

   In the template palette, double-click the action function block type that is to run the new LIN Action. A **Make Action… (*block type*)** dialog appears.
3. Type in a name for the new action and press **OK**. The new blank Action window opens - ready to be configured - and a new entry appears in the **Contents** pane.

20.7.4 Monitor a LIN Action

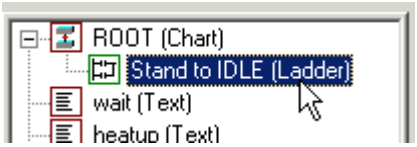
In the Sequence Action Monitor, LIN Actions (Structured Text or Ladder Diagram) can be inspected and monitored as follows:.

1. In the Sequence Action Monitor, pull down the **Online** menu and select **View Action Block…** to show the Choose a Block to Monitor display. This lists the available LIN Actions by Action block name, Action name, and Action filename.
2. Select an Action block and view its associated Action (Structured Text or Ladder Diagram).

20.7.5 Monitor a Ladder Diagram Transition

In the Sequence Action Monitor you can inspect and monitor Ladder Diagram Transitions as follows:

1. In the LIN Sequence Action Monitor, locate the required Transition in the contents pane and double-click it.
   - In the **Contents** pane, Ladder Diagram Transitions appear as sub-windows of the ROOT chart, e.g.



- Alternatively, double-click directly on the Transition bar in the chart - if its location is known.

The single-rung Ladder Diagram representing the selected Transition opens up as a window in the main Monitor display.

2. In the Ladder Diagram, (double-)click any element to open its Element tab, normally at the foot of the window. This shows the element's parameters - **Rung Label**, **Variable**, **Type** - as read-only values.

3. Hover the cursor over or near an element in the Ladder Diagram to reveal a yellow 'tooltip' box indicating the attached Database point and its current value - e.g. **pulse.In = FALSE**.

4. Look at the colours of the different contact and coil elements in the Ladder Diagram. These tell you the current TRUE/FALSE state of the associated variable.

20.7.6 Use the Find utility

In a large configuration it can be difficult to find a particular function block, text string, action, step, etc. This utility helps to locate such objects, as follows:

1. Click the 'Find...' item in the Edit menu. The 'Find...' display appears, containing one or more tabs, depending on the context.
2. Click a tab to display the required page:

- a Block page - finds function blocks.
- a Field page - finds function block fields.
- a Text page - finds text strings in Actions, comments, and ladder diagram fields.
- an Action page - finds chart, ladder, and text Actions.
- an Association page - finds Action associations.
- a Step page - finds Sequence Steps.
- a Tag page - finds Tags.
- a Changes page - finds changes between the Strategy files on the Computer and the 'live' Instrument.

Note: Some pages contain buttons that are identical to the commands available on the context menu.

20.7.7 Save a LIN Action

Saving Action files

When a save operation is carried out in the LIN Action Editor, LINtools saves all the Actions in the current configuration as a pair of files - called filename**.stx**, and filename**.sto**.

- filename is the Action file name entered during the save.

Note: All the current Actions are saved at once as a group - not just the last one edited or created. To exclude any current Actions from the save, they must first be deleted.

Database filename	File Type
<filename>**.stx**	Textual version of all Actions in the file.
<filename>**.sto**	Compiled version of all Actions in the file.

The **.stx** file stores the as-entered structured text for all the Actions that were in the configuration when it was saved. Although the file is not required by the target instrument, it should be kept in case it becomes necessary to recall and edit an Action via the Action Editor at a later date.

The **.sto** file contains the compiled (machine-readable) version of all the text Actions in the .stx file. This file must be downloaded to the target instrument, where it is needed at runtime to execute the Actions enabled by the special 'Action-type' blocks in the control Database.

Note: When saving a LIN Action these two files are combined and stored in the Configuration folder.

To save LIN Actions:

1. To save the Action files to their existing path/filename, click the Save toolbutton (or use the **File** > **Save** pulldown).
2. To save the Action files to a new path/filename, select **File** > **Save As...**, and enter a new filename.
3. The Actions are automatically compiled as well as saved, and a **Compile Results** report is displayed in a separate pane.
4. Trace and correct any errors using the compile report, and then re-save the files.

Note: If more convenient, error-correction can be postponed to a later editing session. The Actions are saved regardless.

20.8 EDIT A LADDER DIAGRAM

Ladder Diagrams are configured:

- in the Sequence Editor, if the Ladder Diagram is to be a Sequence Action or Transition, or
- in the Action Editor, if the Ladder Diagram is to be a LIN Action.

In either case the Ladder Diagram configuration procedure is the same:

1. Open the appropriate editor.
   - For LIN Actions the user must select an Action block type from the Template Palette to display the 'Make Action' dialogue (see also the next step).
2. If necessary, click the Make Action toolbutton to start the Make Action wizard and display the Make Action dialogue.
3. Select the required Action type radio button and complete each section of the Wizard. Enter a **Name** for the action. A blank Ladder Diagram editing window is displayed.

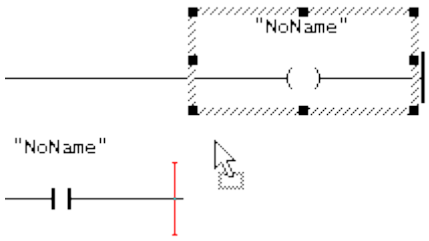
Note: Additional information is displayed if the Help checkbox is set  .

4. Select a Ladder element by clicking on one of the eight Ladder Diagram toolbuttons, found usually at the top-right of the editor window. This 'loads' the cursor with the selected element (as shown by the cursor format adopted). If the Function toolbutton is selected, it may be necessary to select a function type from the Template Palette ('Functions' tab). Double-click the required function to 'load' the cursor.
5. Locate the loaded cursor in the Ladder Diagram editor window at the required insertion position for the element. A red insertion marker appears if the chosen location is valid. Otherwise, a grey insertion marker appears with a 'prohibited' sign. Click to paste the element, which then automatically 'connects' into any existing elements.

   If the cursor is to remain loaded with the current element, the <Ctrl> key should be held down during the paste operation.
6. Continue building up the Ladder Diagram by selecting and placing elements and associating Variables.

   - Existing elements can be dragged to new positions in the diagram (hold down the left mouse button), or they can be deleted, cut, copied, and pasted using 'Edit' menu items or

the standard 'Windows' shortcuts. Edits can be 'undone' or 'redone' using standard 'Undo' / 'Redo' toolbuttons.



20.8.1 Parameterising the Ladder Diagram

Placed elements appear initially in their default format, and associated with the default '**NoName**' field. Each element and rung in the Diagram must be edited and associated with a database field as appropriate.

To parameterise a Ladder Diagram

1. Click (or double-click) an element to reveal the appropriate elements details, in the Object properties pane (the white fields are editable).
2. Type in an (optional) 'label' for the rung in the 'Rung Label' box. Labels appear in blue at the left of the diagram above the relevant rung.
   - Labels act as targets for jump instructions. They can have up to 16 characters including **0-9**, **A-Z**, **a-z**, and _ only.
   - The label - **_End** - appears after the last rung by default.
3. Associate the element with a Variable (for contacts, coils, and variables only).
   - Double-click the **Variable** field to see a button. Click the icon to display a Picklist of all available fields, then double-click the required field. The Picklist closes and the selected field appears in the '**Variable**' box.
   - Alternatively, select the required '**Variable**' from the '**Variables**' tab on the Template Palette and move (drag) to the element requiring an associated element.
4. To edit the element's 'type' (contacts and coils only), click the **Type** white box and select a type from the pulldown menu. The element's icon changes in the Ladder Diagram.

20.8.2 Associate a Variable

Variables can be associated with,

- Ladder Diagrams elements in either a LIN Sequence or a LIN Action
- an alias for a Generic name in the Sequence Map table
- Structured Text of an action
- LIN Reference in the Profibus Configurator

To associate Variables:

1. Open the Palette. Press **View** > **Palette** to reveal the Palette pane.
2. Select the required **Variable** from the **Variables** pane on the Template Palette and move (drag) to the element/LIN Reference requiring the variable.

Note: The Variable can also be typed directly in a field, using the block.field convention.

- In a LIN Sequence double-click the **Variable** field in the **Object Properties Pane** to display the available Variables.

Notes:
1 Pressing the PC's <F9> key displays the Variable picklist in a LIN Sequence. A filtered picklist may be expanded to show all available variables by pressing the <F8>.
2. Generic Sequence steps are displayed when associating Variables to Ladder Diagram elements.

- In the Map table, double-click an **Alias for...** field to enter it into the **Map** table. Subfield context menus are handled in the same way.

20.8.3 Save a Ladder Diagram

If the Ladder Diagram is a LIN Action, created in the Action Editor, it can be saved in the same way as any other LIN Action.

Saving a LIN Action

If the Ladder Diagram is a Sequence Action or Transition, created in the Sequence Editor, it can be saved in the same way as any other Sequence Action or Transition.

Saving a LIN Sequence

20.8.4 Compile an Action or Transition

Compiling an Action or Sequence Transition converts it to 'object code'. If LINtools encounters any errors during an attempted compilation it reports them in a 'compiling' pane. Compiling is, therefore, a good way to check the validity of Structured Text or ladder diagrams.

To compile a single Action or Transition:

1. Right-click in the Action window or transition window and select the **Compile [...]** item from the menu. A new pane appears reporting the results of the compiling operation. Any errors found are listed.

   - Alternatively, in the **Edit** menu, click **Compile [...]**.

Note: Compiling does not save the object code - only File > Save does. But if you do save the Sequence or actions file you will see a compile report for all the Actions in the file.

2. Double-click an error line in the report to highlight in the action window or transition window the portion of structured text or ladder diagram that needs correcting. (The relevant Action window or Transition window appears if not already on view.)

   - Sometimes in a ladder diagram an error relating to the **whole** rung (e.g. 'Bad label') highlights the **end** of the rung. Similarly, an error due to a fault in the **body** of the diagram (e.g. 'Unresolved jump') is only highlighted at the **last item** in the diagram.

3. When all errors have been corrected, recompiling the Action or Transition produces a **'No Errors'** report.

Note: If a generic sequence is being configured, but the map is incomplete, any 'Can't find this name' errors can be ignored, as these errors will be corrected when a specific sequence is generated. Other structured text errors should be corrected now.

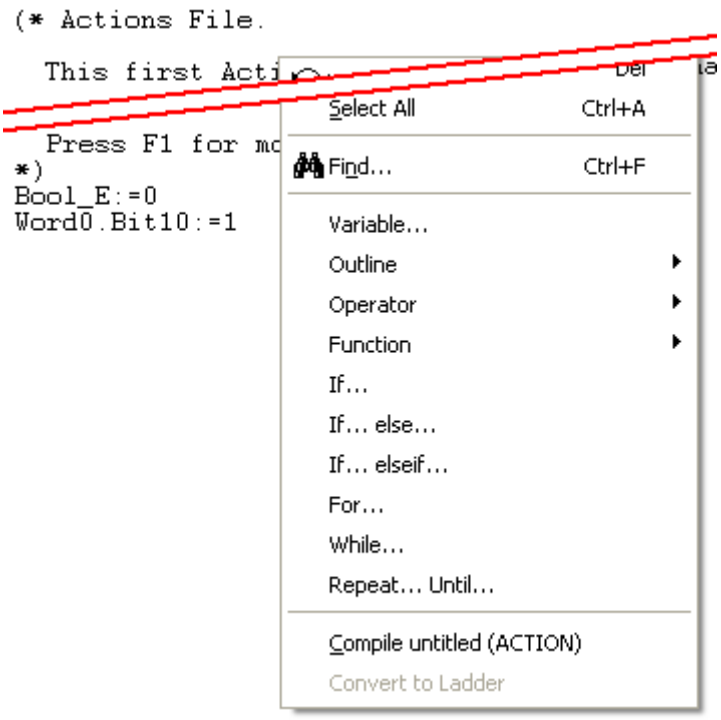
20.9 CONFIGURE STRUCTURED TEXT

20.9.1 Edit Structured Text

Assuming the Structured Text window is open, right-clicking in a Structured Text Action window or Transition tab reveals a context menu that allows the Structured Text items to be selected for use.

To edit Structured Text (ST)

1. Associate a Variable to define the database block field to be configured using the Variable command.



2. Select required default Structured Text syntax. Use the required command to insert the default syntax at the selected position to help you write Structured Text.

Note: Use these commands to prevent syntax errors.

3. Compile the Structured Text to check the validity of Structured Text and report any errors in a **Compile Results** pane

20.9.2 Associate a Variable

Variables can be associated with,

- Ladder Diagrams elements in either a LIN Sequence or a LIN Action
- an alias for a Generic name in the Sequence Map table
- Structured Text of an action
- LIN Reference in the Profibus Configurator

To associate Variables:

1. Open the Palette. Press **View** > **Palette** to reveal the Palette pane.
2. Select the required **Variable** from the **Variables** pane on the Template Palette and move (drag) to the element/LIN Reference requiring the variable.

Note: The Variable can also be typed directly in a field, using the block.field convention.

- In a LIN Sequence double-click the **Variable** field in the **Object Properties Pane** to display the available Variables.

Notes:
1. Pressing the PC's <F9> key displays the Variable picklist in a LIN Sequence. A filtered picklist may be expanded to show all available variables by pressing the <F8>.
2. Generic Sequence steps are displayed when associating Variables to Ladder Diagram elements.

- In the Map table, double-click an **Alias for...** field to enter it into the **Map** table. Subfield context menus are handled in the same way.

COMPILE STRUCTURED TEXT

The Compile operation checks the validity of Structured Text and reports any errors in a "Compile Results' pane. Each transition or action containing an error is listed on a separate line of the report, together with the (first instance of) offending text, and the type of error.

Note: Select 'View' > 'Compile Report' to show and hide the Compile Results pane as required.

Structured Text can be compiled in several ways:

- Structured Text can be compiled in an individual Action or Transition.

   Compiling an Action or Transition
- For Transitions, all the Transitions in an SFC can be compiled in one operation.

   Compiling all SFC Transitions
- All the Structured Text in the Sequence is automatically compiled whenever the Sequence is saved, or if File>Compile is selected.

   Saving a LIN Sequence

20.9.3 Compile an Action or Transition

Compiling an Action or Sequence Transition converts it to 'object code'. If LINtools encounters any errors during an attempted compilation it reports them in a 'compiling' pane. Compiling is, therefore, a good way to check the validity of Structured Text or ladder diagrams.

To compile a single Action or Transition:

1. Right-click in the Action window or transition window and select the **Compile [...]** item from the menu. A new pane appears reporting the results of the compiling operation. Any errors found are listed.

   - Alternatively, in the **Edit** menu, click **Compile [...]**.

Note: Compiling does not save the object code - only File > Save does. But if you do save the Sequence or actions file you will see a compile report for all the Actions in the file.

2. Double-click an error line in the report to highlight in the action window or transition window the portion of structured text or ladder diagram that needs correcting. (The relevant Action window or Transition window appears if not already on view.)

   - Sometimes in a ladder diagram an error relating to the **whole** rung (e.g. 'Bad label') highlights the **end** of the rung. Similarly, an error due to a fault in the **body** of the diagram (e.g. 'Unresolved jump') is only highlighted at the **last item** in the diagram.
3. When all errors have been corrected, recompiling the Action or Transition produces a **'No Errors'** report.

Note: If a generic sequence is being configured, but the map is incomplete, any 'Can't find this name' errors can be ignored, as these errors will be corrected when a specific sequence is generated. Other structured text errors should be corrected now.

20.9.4 To compile all the Transitions in an SFC

1. In the SFC worksheet, right-click anywhere to reveal a context menu. Select the **Compile [name] (Chart)** item. The results of the compilation are listed in a report pane.
2. Double-click an error line in the report to display the invalid transition text in a separate Transition tab, where it can be corrected.
3. When all errors have been corrected, recompiling the SFC produces a **'No Errors'** report.

Note: If a generic sequence is being configured, but the map is incomplete, any 'Can't find this name' errors can be ignored, as these errors will be corrected when a specific sequence is generated. Other structured text errors should be corrected now.

20.10 USE THE ONLINE RECONFIGURATION

20.10.1 Online Reconfiguration - Overview

Online Reconfiguration is used for editing an existing Strategy currently operating in a 'Live' LIN instrument. It allows the user to make changes (ranging from individual block parameters to a complete Strategy change) while the plant/system is still running, therefore, reducing the overall downtime of the plant/system.

The functions for Online Reconfiguration are similar to editing an offline Strategy, but additional functions permit the user to Try (test) the changes in the 'Live' Instrument before all changes are finally Applied or discarded if the changes are not required. If changes exist between the 'Live' LIN Instrument and the LIN Database on the Computer, the 'Find' dialog can be used to locate the changes that match the configured criteria.

The Connect feature allows communications with a specific Instrument. Once LINtools has connected to the LIN instrument, it can be reconfigured or will allow a limited degree of control via the Instrument Status dialog.

Connection to a 'Live' LIN Instrument is via the Online Reconfiguration toolbutton on the toolbar or **Online** > **Reconfigure** in the Menu bar. However this function is enhanced with the addition on the following toolbuttons.

- Try toolbutton

Note: Operating this toolbutton again while Trying the strategy cancels the operation, returning to the Online Reconfiguration mode with all existing values.

- Apply toolbutton

Note: Edited values are discarded using the Online > Discard Changes command.

20.10.2 Connect to an Instrument

Connecting to an Instrument allows the 'Live' Instrument Strategy files to be captured in readiness for monitoring or reconfiguration. Once the Strategy files have been captured, they can be edited either online (after the Online Reconfiguration button is clicked on), or offline, and then downloaded to the LIN Instrument. When connected, selected Watch lists can be created and saved via the **Online** > **Watch Window Setup** command, allowing specific LIN Database block fields to be monitored.

This feature is useful when the system already exists but the Instrument Folder in the Project is yet to be generated.

Note: If a LIN Database is already open in LINtools it is used, otherwise, the Strategy is captured in memory from the target LIN Instrument.

TO CONNECT TO A 'LIVE' INSTRUMENT

1. With LINtools open and without a LIN Database file, click on the Connect toolbutton.

   - Alternatively, select **File** > **Get me started** to run the wizard, then select the **Connect Online without files** radio button.

   Both offer the chance to locate the required Instrument by selecting the Network and then the Instrument itself.

   Note: If a database is not open in LINtools, a dialog is displayed. Use this dialog to specify the Port and Node of the instrument required.

2. Once connected, as indicated by the Connect watermark and Connection Indicator, clicking the Online Reconfiguration toolbutton allows the 'Live' Instrument Strategy to be edited. Clicking the Connect toolbutton again allows the captured Strategy files can be edited offline.

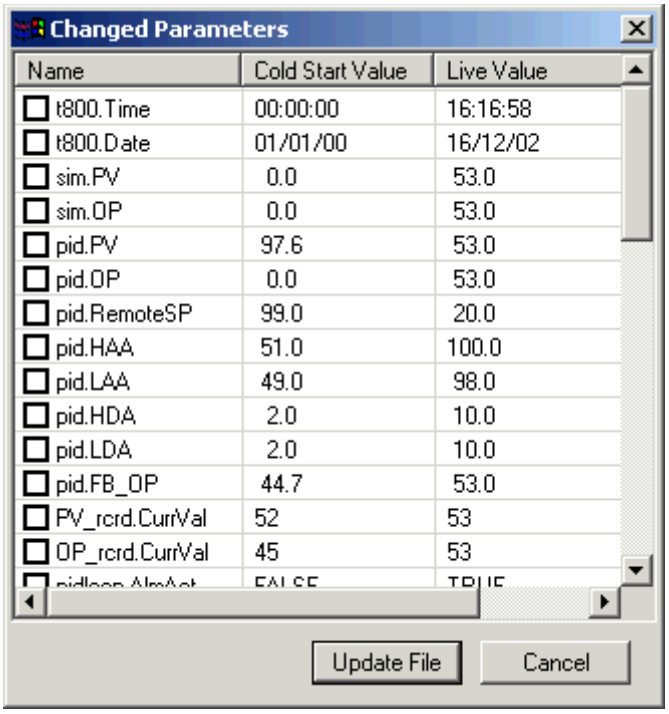
Note: Clicking on the Save toolbutton, after first disconnecting the 'live' Instrument will save the Strategy files to the current Project.

20.10.3 Scan for changed parameters

While monitoring a remotely running LIN Database the user can scan the Database to see if any field values have changed from their initial cold-start values. If they have, selected changed values can be captured and uploaded to the LIN Database, overwriting the original values.

To scan a Monitored Database:

1. Connect to an Instrument and then, from the **Online** pulldown menu select **Scan For Changed Parameters**. A message announces that the scan has started.
2. When the scan has finished a **Changed Parameters** window opens, listing the fields that have changed, with their 'cold start' and 'live' values - see image below:



3. Review the changed fields and select those to be uploaded by clicking their checkboxes. Then click the 'Update File' button to start the upload. Progress is reported in an information pane, which shows the number of successfully updated fields, and also warns if any fields have failed the update process.

   Each updated field is verified against the snapshot 'live' value. A write-fail occurs if the live and written values are not the same for any reason - e.g. if the fields are read-only.
4. If all selected fields are written successfully, the 'Changed Parameters' window closes. Otherwise it remains open with the already written fields deleted from the list.

Note: Due to the time it takes for all the live values to be scanned in the remote instrument, the values are not all recorded at the same time. Parameter scanning is most useful for fields that change slowly, e.g. tuning parameters.

20.10.4 Use the Watch window

The **Watch Window** facility allows up to 100 remote LIN Database fields to be selected for continuous monitoring (and editing) in a special window. Only one watch window can be displayed in the Monitor window at a time, but Watch windows can be configured and saved, and subsequently opened as the current window.

To open/close the current Watch window:

Click the Watch toolbutton (or select **Online** > **Watch**).

- The current Watch window opens, or closes if already open.
- You can also close the Watch window by right-clicking its banner and selecting **Hide**.
- Adding a field to the current Watch window

You can add a field to the current watch window:

- directly via the Watch window
- from the required block's Block tab field
- via the Find Field dialog

An attachment to the remote node is formed for a cached block field. The added field name appears under the **Name** column of the Watch window, its value under the **Value** column, and a Description (if any) under the **Description** column.

Note: The Watch window opens (if not already open) when a field is added to it.

To remove a field from the current Watch window:

In the watch window, right-click on the field and select **Remove Watch** from the context menu.

To move and copy fields in the current Watch window:

Locate the cursor over the field to be moved and hold down the left mouse button. Drag the field to the row where you want it moved, then release the mouse button to 'paste' it in position.

- To leave the original field in position - i.e. make a copy - hold down the PC's <**Ctrl**> key before you release the mouse button.

To highlight the block associated with a watched field:

Right-click the field in the watch window and select **Open Block**.

- The associated block highlights in the Function Block Diagram, and its **Block** tab appears.

To change the displayed resolution of a watched analogue field:

Right-click anywhere in the field's row and click **High Resolution** to toggle the item's display resolution.

- By default the watch window displays floating-point numbers with a precision similar to the instrument front-panel displays. Selecting high resolution format displays them with maximum precision (7 significant figures).

To edit the value of a watched field:

- **Numeric fields**

Click a numeric **Value** field once to insert an edit cursor, and edit the field to the required value. Double-clicking highlights the whole field.

- **Enumerated fields**

Click the field to see a picklist, and select the required value from the menu.

Note: When an editable field is clicked on, a yellow 'tooltip' appears over the Units cell showing the field's current value.

To clear the current Watch window:

Right-click the window's banner and select **Clear** from the menu.

- Alternatively, select **Online** > **Watch Window Setup** > **Clear.**

Note: The current watch window clears - whether open or closed.

To save the current Watch window setup:

1. If the current Watch window is open, right-click in its banner and select Save... from the menu. Alternatively, or if no watch window is open, in the Online > Watch Window Setup > Save....
2. Fill in the Save (Watch window) dialog.

To open a saved Watch window:

1. If the current Watch window is open, right-click its banner and select Open... from the menu. (Alternatively, or if no watch window is open, select Online > Watch Window Setup > Open....)
2. Fill in the Open (Watch window) dialog.

Redo All Attachments

If attachments to any viewed points fail, the **Redo All Attachments** option attempts to restore all the attachments.

Example

If you access the LIN Monitor Database when an instrument is disconnected from the network, no attachment will be made even after subsequent reconnection - until this option is selected.

To redo all attachments:

Select Online > Watch Window Setup > Redo All Attachments.

OPEN / SAVE WATCH WINDOW DIALOGS

Use these dialogs to open a new/current Watch window, or to save the current watch window.

To access the dialogs, right-click in the Watch window banner and select **Open...** or **Save...**, respectively, from the menu.

- Alternatively - if no Watch window is open - in the **Online** menu, select **Watch Window Setup** > **Open...** or **Save...**.

To open a saved Watch window:

In the **Open** dialog, click on a Watch window setup icon (or type its name in the **Name** box) and hit **Open**. The dialog closes and the selected watch window opens, replacing any currently-opened window.

To save the current Watch window setup:

In the **Save** dialog, type a name for the window in the **Name** box and hit **Save**. The dialog closes and the current watch window is saved under the specified name.

CHOOSE A BLOCK TO MONITOR

Use this dialog to select an Action block type and its associated LIN Action to monitor in the Sequence Action Monitor window.

To access the dialog:

In the Monitor window, in the **Online** pulldown menu select **View Action Block…** to show the **Choose a Block to Monitor** dialog.

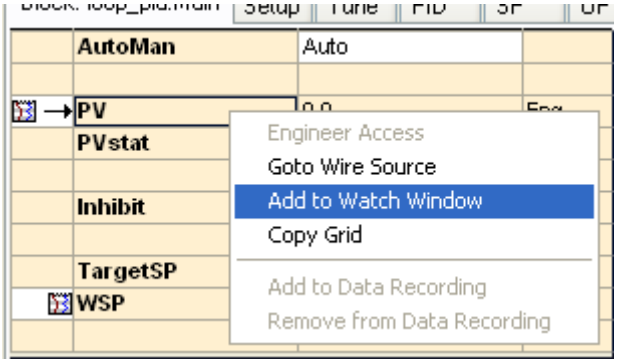
Use the dialog

The dialog lists all the Action-type function blocks in the LIN Database by name - and their corresponding Action names and the filename(s) of the .sto Action file(s) in which they are stored.

Click an action function block name to highlight it, then click **OK**. The Action (Structured Text or Ladder Diagram) appears in a sub-window, ready for monitoring.

ADD A BLOCK TAB FIELD TO THE WATCH WINDOW

This item is active in Connect (Monitor) mode.



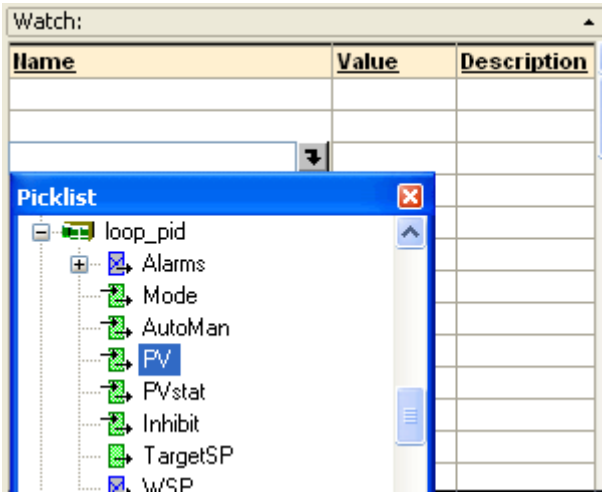
To add a Block tab field to the Watch Window:

Right-click anywhere on the field to reveal a context menu - see above - then click the **Add to Watch Window** item.

When first adding a field to the Watch window, the Units of the field are copied into the Description. This field will not change dynamically when editing the units of the field in the database.

Note: To update the Description field remove the field from the Watch window and exit Connect mode. Change the Units field while offline and download the file to the instrument. Then add the field to the Watch window again.

ADD A FIELD DIRECTLY TO THE WATCH WINDOW



To add a field directly to the currently-open watch window:

1. Double-click a **Name** cell in the row where you want the new field added. A button appears at the right of the selected cell.

2. Click the button to see a menu of available fields, arranged in Windows 'tree' style. Double-click a field. The menu closes and the selected field appears in the watch window.

- You can also **overwrite** existing fields in this way or type the *block.field* as required.

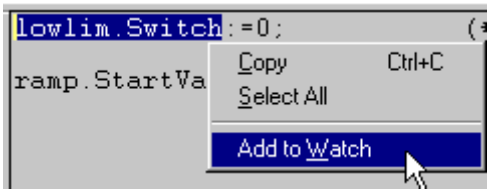
Note: If the *Options.CommsDis* bit of the instrument Configuration (Header) block is set TRUE, UNCONFIRMED field writes are not permitted. UNCONFIRMED field writes result from connections into cached blocks. Values written using LINtools while online and connected to the instrument are CONFIRMED field writes and are therefore unaffected by this bit.

When first adding a field to the Watch window, the Units of the field are copied into the Description. This field will not change dynamically when editing the units of the field in the database.

Note: To update the Description field remove the field from the Watch window and exit Connect mode. Change the Units field while offline and download the file to the instrument. Then add the field to the Watch window again.

ADD AN ST BLOCK FIELD TO THE WATCH WINDOW

When monitoring a Sequence, you can add LIN function block fields - appearing as Structured Text in Transition or Action windows - to the Watch Window.



To add an ST block field to the Watch Window:

1. Using the text cursor, highlight the complete field - including the block name but **excluding** any ST operators or other characters.
2. Right-click in the window and click **Add to Watch** in the menu. The selected field appears in the Watch Window.

20.10.5 Access the Online Reconfiguration

Online reconfiguration permits the user to make changes to the database of a Strategy that exists on the Instrument currently active in the plant/system.

1. With LINtools Engineering Studio open, press the Online Reconfiguration toolbutton.
   - Alternatively, select **Online** > **Reconfigure** from the Menu bar, or use select **File** > **Get me started** to offer the option to connect to a LIN Instrument that is not open.

Note: While accessing a 'live' LIN Instrument, LINtools will illuminate a small indicator in the lower right corner of the Status bar (

).

2. The message '**Making attachments to LIN OPC Server. Please wait…**' appears.
3. After a delay the FBD including any Online Reconfiguration changes appear with a online reconfiguration watermark indicating that LINtools is currently online.

Note: Under certain LINtools customised colour schemes, watermarks can become obscured. The intensity of the watermark can be adjusted by changing the Watermark intensity value via the View > Options > Colour dialogue.

At this point the LINtools program compares the instruments latest structural edits to assess the Alignment Options.

- If differences between the LINtools and target instrument Strategy exist a prompt appears. Press **OK** to continue.

4. Selecting 'Apply' updates both the LINtools and target instrument Strategy with all current Online Reconfiguration changes. If 'Discard' is selected the changes are ignored and LINtools will show the target Instrument LIN Database.

- Pressing the **Online Reconfiguration toolbutton** again, reverts to offline editing.

Customise the default colour coding

20.10.6 Reconfigure an online strategy - Overview

The following stages may occasionally overlap, and their order can be varied when editing an existing Strategy. Function blocks, wiring, parameter values and Sequences can be added, modified, or deleted at any time using the Online Reconfiguration facility.

Note: The Online Reconfiguration uses the same commands as the LIN Database Editor with the addition of the commands mentioned in the section.

1. Open and ensure the required LIN Database is the focus (selected file).

Note: If the Strategy files does not currently exist on the Computer use the Connect toolbutton to locate the required instrument on a network.

2. Click on the Online Reconfiguration toolbutton. 

   - Alternatively, select **Online** > **Reconfigure** from the menu bar.

   At this point the LINtools program compares the instrument's latest structural edits to assess the Alignment Options.

   - If configuration changes exist, a prompt appears describing the Online Reconfiguration options available.

3. Edit the required strategy as appropriate. Use the same LIN Database instructions for editing both Offline and Online strategies.

   i. To be able to undo changes to parameters instantly, first Unlink the function block.

Caution

When a block is unlinked, not all data from a target instrument Strategy is captured. Unlinking the block replaces the existing block with the 'Tentative' unlinked block containing the 'Cold Start' values. If live target instrument data is required use the 'Scan for changed parameters' command.

   ii. Make any changes required. These are 'Tentative' changes, and do affect the target instrument strategy.

   iii. Press the 'Try' toolbutton to test the target instrument Strategy.

   If satisfied with the changes, press the 'Apply' toolbutton to ensure both LINtools and Target instrument strategies are the same.

   Configure a LIN Database

Note: The physical properties of I/O blocks can be changed only after the block has been Unlinked. Once changes are complete the block can be Re-linked.

4. Try the changes in the target instrument. When the attempts at achieving a satisfactory strategy are complete, Untry the changes.

Note: Tries can continue while LINtools is Offline by exiting Online Reconfiguration without selecting the Untry command. However, when next entering Online Reconfiguration, the previously existing changes appear after a prompt warning of changes to the instrument strategy.

   - Repeat the previous instructions until satisfied with the complete control strategy.

Note: Edited values may be discarded using the Online > Discard Changes command.

5. When satisfied, Apply all the changes to the live instrument Strategy. This ensures that all previously 'pending' changes are saved directly both to the live instrument Strategy and to the LINtools strategy.

20.10.7 Try an Online Strategy

Trying an Online Reconfigured Strategy tests all 'pending' changes in the live instrument Strategy. The results of the values entered in the Strategy can be assessed before applying them to the system.

Note: Under certain LINtools customised colour schemes, watermarks can become obscured. The intensity of the watermark can be adjusted by changing the Watermark intensity value via the View > Options > Colour dialog.

1. With Online Reconfiguration operating and all required values changed as appropriate, press the Try toolbutton.

   - The **Try** command is indicated by a change of watermark after momentarily displaying a '**Try Changes**' dialog.

Note: Tries can continue while LINtools is Offline by exiting Online Reconfiguration without selecting the Untry command. When next entering Online Reconfiguration appropriate prompts appear warning of changes to the instrument strategy and asking the user to select a strategy for operation in the instrument.

   - Clicking on the **Try** toolbutton again, cancels the **Try** operation after momentarily displaying an '**Untry Changes**' dialog. The watermark reverts to the Online Reconfiguration indicator.

 Customise the default colour coding

20.10.8 Apply an Online Strategy

Applying an Online Reconfigured Strategy confirms that all 'pending' changes to a live Strategy in the system are as the user requested.

Caution

The applying of online Reconfiguration changes is irreversible.
Before applying online reconfiguration changes it must be ensured that the strategy is operating as required, or the system may suffer serious consequences as a result.

1. With Online Reconfiguration operating and all values changed as appropriate, press the Apply toolbutton.

   - The Apply command momentarily displays an '**Apply Changes**' dialog.

2. Finally the **Save** dialog appears.

   Saving Online Reconfiguration edits has different results depending on the current state of LINtools.

   - The target instrument will not upload the changes if the Online Reconfiguration is cancelled without **Trying**. Therefore a **Save** dialog is not displayed and the PC version will remain unchanged.
   - The target instrument will upload the changes if the Online Reconfiguration is cancelled after **Trying**. However, this does display the **Save** dialog, which results in either,
   - Selecting **No** from the **Save** dialog will not save the changes to the PC.
   - Selecting **Yes** from the **Save** dialog will save the changes to the PC and display the Build window revealing any errors or warnings that have been created when making the target instrument Online Reconfiguration changes.

Note: LINtools Online Reconfiguration performs only a local Build. It is recommended that a full build is performed to ensure all errors and warnings are displayed in the Build window.
Failure to do so may result in for example, the deletion of cached blocks referenced in other instruments, or which are being displayed on a User Screen.

20.10.9 Unlink a block

Unlinking an Online block permits the configuration of block parameters that cannot be changed in the live block, such as,

- parameters often used at start up of the database
- parameters used to configure other sub-systems
- physical properties of I/O blocks, i.e. the Module Type, SiteNo and Channel fields

An Online block can be unlinked either,

- Automatically

  Any blocks saved in the LINtools Strategy differing from the instrument strategy are detected when the Online Reconfiguration toolbutton  is pressed. These become unlinked blocks and are considered as Offline.

- Manually

  While performing Online Reconfiguration, the user can choose to unlink the block in the running Strategy via the context menu.

  Before an Online block is Unlinked, the inspected block displays live instrument data. When the block is Unlinked from the running target Strategy, it displays the data that exists within the LIN Database, .dbf.

Note: To capture live values from the Instrument, use the Online > Scan for Changed Parameters command. This command compares and displays values, allowing the required value to be configured in the appropriate fields.

To unlink an block:

Assuming that the block is currently Online,

1. Select the block.
2. Right-click to display the context-sensitive menu and select the Unlink command.

   The block is now Unlinked, as indicated by a change in block colour.
3. Edit the block as required.
4. When changes to an **Unlinked block** are complete,

   - Try the changes to ensure the required results.
   - If completely satisfied Apply the changes to the live instrument strategy.
   - **Re-link** the block to cancel changes made specifically to this block.
   - Select **Online** > **Discard Changes** to cancel ALL changes, including changes to other unlinked blocks, since the last save.

To Re-link an Unlinked block

1. To **Re-link** the block, right-click to display the context menu and select the Re-link command.

   The block is now linked, as indicated by a return to the original block colour.

Note: All changes made to the Unlinked block are cancelled if the Re-link command is selected.

 Customise the default colour coding

20.11 PROFIBUS CONFIGURATION

20.11.1 Use the Profibus Master Configurator

The Profibus Master Configurator is used to declare the Profibus Slave devices and the installed modules. Parameters from the LIN database of an instrument operating as the Profibus Master are referenced in the relevant Profibus Slave pages of the Profibus Master Configurator. Placing a LIN database parameter in a LIN Reference field on a particular page determines whether the parameter is written to (Output data) and/or read from (Input data) a Profibus Slave device, at either cyclic or acyclic data exchange.

The Profibus Master Configurator uses a tree view structure to define the Profibus Network and Profibus Slaves, in alphabetical order. Profibus Slaves can be Compact devices or Modular stations. Any Modular station will require configuration of each module fitted the device.

Note: Each page that is made accessible when the Profibus Master is selected displays particular information related to the Profibus Master, including currently available address space for the Input data, Output data and Acyclic data, and currently used space for Extended Diagnostics.

To use the Profibus Master Configurator,

1. Configure the LIN Profibus GateWay.

   The LIN Profibus GateWay is the interface between the LIN instrument operating as the Profibus Master and the devices communicating via the Profibus Network connected to this Profibus Master.

2. Configure the Profibus communications protocol for the Profibus Master.

   The COM port Protocol is configured on the Instrument Option page of the Instrument Properties dialog. It defines the communications Protocol used by the selected COM port, i.e. Profibus Master COM port should be configured to ProfibusDvp1-M.

Note: The Instrument Options in the Instrument Properties dialog of the Profibus Master must be configured to use the Profibus Protocol, ProfibusDvp1-M.

3. Configure the Profibus Slave, as required.

   Each device is added using the New Instrument wizard. The wizard is used to define the Profibus Slave folder properties, including the folder name, Instrument type, and Slave address. Once completed, the .gsd file must be defined to ensure the Profibus Master can understand what data is expected. Modules can be added for each modular device using the **Insert** button, and finally, the device data exchange can be configured.

Note: The New Instrument wizard supports the use of devices that are unknown by the Configuration system. Unknown devices can be selected using the '3rd-party' option and defined using the relevant .gsd file. These files can be imported to the appropriate GSD library when required.

4. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each operation it attempts and any warnings or errors that may have been detected. This also generates the GateWay file, .gwf, and Profibus Binary file, .upb, and automatically adds them to the list of Files to be Downloaded.

5. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

   The Instrument configuration contains all the files used by the Instrument to perform its required function.

20.11.2 Configure the LIN Profibus GateWay

The LIN Profibus GateWay provides a communications interface between an Instrument on a Local Instrument Network, LIN, operating as a Profibus Master and a number of third-party devices on a Profibus network.

The interface between these communications protocols is defined via a GWProfM_CON block in the LIN database of the LIN Instrument operating as a Profibus Master. Configuration of this block identifies a specific LIN Profibus Master Configuration file, .upm, GateWay file, .gwf, and Profibus Binary File, .upb.

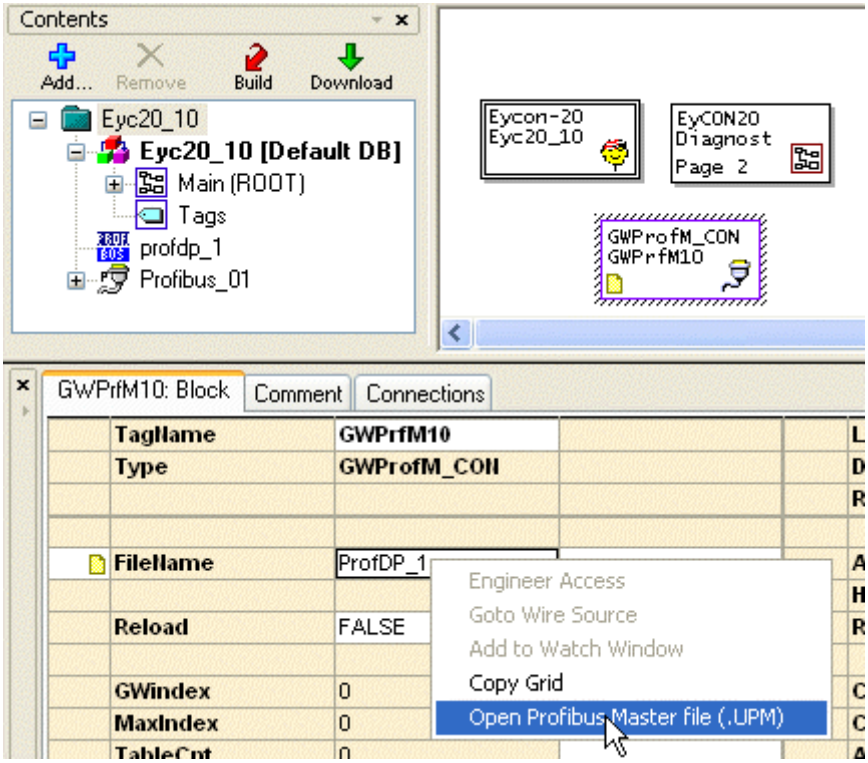
To configure the LIN Profibus GateWay:

1. Open the Function Block Template Palette, and expand the Comms category to reveal a list of compatible Comms function blocks, including the GWProfM_CON.
2. Select and drag the GWProfM_CON block to the worksheet to create a copy of it.
3. Open the Object Properties Pane to show the block fields, and name the block.

Note: A indicates that an additional file is associated with this block and the filename must be entered.

4. Select the FileName field and enter a name, 8 characters maximum i.e. ProfDP_1, for the required LIN Profibus Master Configuration file, .upm. If the required LIN Profibus Master Configuration file, .upm, already exists, enter this filename.

Note: Try using a version of the Profibus Network folder name. This can simplify the identification of the related GateWay file, .gwf, and Profibus Configuration file, upm in a specific LIN instrument.



Note: Use the Build process liberally to ensure all information is accessible to other applications.

5. Select the FileName field to show the context menu, and select the **Open Profibus Master file (.UPM)** command.

   If the LIN Profibus Master Configuration (.upm) file does not exist, this displays a dialog used to create the .upm file with the name configured in the FileName field. Press Yes to continue.

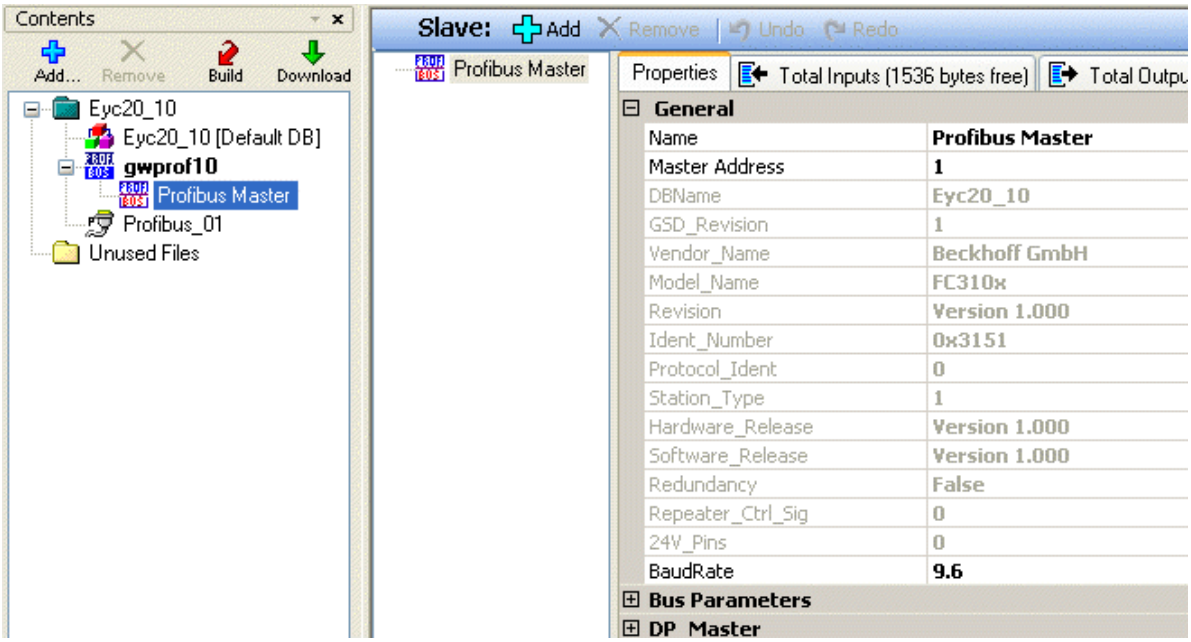
   The Network wizard appears.

Note: Help relating to each specific page of the wizard is shown when the Help checkbox is set .

6. Complete each section of the wizard to create a new Profibus Network or determine which existing Profibus Network will be used. Follow the instructions displayed in the wizard.

7. Press the **Finish** button to complete the wizard, and automatically launch the Build process. This will add the configured files to the Project.

   When the Build process completes, LINtools launches the Profibus Master Configurator, automatically defining the Instrument containing the GWProfM_CON block as the Profibus Master.



20.11.3 Configure the Profibus communications protocol

Notes:
1. If the communications protocol is not configured correctly, the Profibus data exchange will fail.
2. The instrument must be online before the download of the Instrument Option parameters can start.

The Communications Protocol is used to define the Network Protocol connected to the specified COM port to be used for communications with the devices present on the network.

The COM port Protocol is configured on the Instrument Option page of the Instrument Properties display. It defines the communications Protocol used by the selected COM port, i.e. Profibus Master COM port should be configured to ProfibusDvp1-M.

20.11.3.1.1 To configure the Communications Protocol

1. Right click on the LIN Instrument Folder in the LINtools Contents pane to show the context menu.

2. Click on the 'Instrument options' item. The Instrument Option page shows configuration data for various aspects of the instrument, e.g. HMI settings, Modbus or Profibus ports configuration, etc..

   i. Select the **Profibus** category, to display the Hardware and Protocol parameters associated with Profibus communications.

   ii. Press the **Upload current options settings** button. This will connect to the instrument and update the instrument hardware configuration used by LINtools.

   The message '**Making attachments to LIN OPC Server. Please wait…**' appears.

iii. When eventually connected, the Protocol parameter for the corresponding COM port, e.g. PROF 1, should show ProfibusDvp1-M. This configures the PROF1 COM port of this LIN Instrument for use with ProfibusDvp1-M protocol.

If the Protocol parameter does not show ProfibusDvp1-M, select it from the Picklist.

iv. Save all changes. If the changes are made to an instrument, already within an instrument folder, a download process is initiated via the 'Download new settings' dialogue.

If the 'Download new settings' dialogue appears, press Yes to attach to the instrument, and automatically download the changes to the instrument. Press No to continue configuring the Instrument Option parameters.

20.11.4 Configure the Profibus Master Properties

Configuration of the Profibus Master Properties parameters consists of editing the General Profibus Master device details, Bus Parameter (Network) details and Profibus Master Data exchange limitations. To configure the Properties Master,

1. Select a Profibus Master to reveal the Properties page.

   The parameters on this page relate to features that are supported by the Profibus Master. Only the parameters for the selected Profibus Master are edited when any parameter on the Properties page is changed.

   Most device Properties page parameters remain read only, but some can be edited by selecting a value from a Picklist, changing the module type, or by changing the value.

Note: The .gsd file relating a Profibus device can be edited using the Profibus GSD File Editor at Start > Programs > … > Profibus GSD File Editor where '…' denotes the installation path

2. Edit the parameters on the Properties page, as required.

   - Configure the General parameters. This section refers to the device.
     If this section of parameters are condensed, press the '⊞' button to show these parameters. The section can be condensed using the '⊟' button.

Note: Select the highest Baud Rate value that does not cause the Warning message to show that an unsupported Baud Rate has been configured in the Profibus Master.

   - Configure the Bus Parameters. This section refers to the operation of the network.
     If this section of parameters is condensed, use the '⊞' button to display them. The section can be condensed using the '⊟' button.
   - The DP_Master parameters are read-only and refer to the data exchange in the Master.
     If this section of parameters is condensed, use the '⊞' button to display them. The section can be condensed using the '⊟' button.

3. Configure the Profibus Slave.

   Although the .gsd file provides detailed information about the Profibus Slave, the module configuration will not be known, and will have to be configured.

4. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each attempted operation and any warnings or errors that have been detected. The Save operation also generates the GateWay file (.gwf), and the Profibus Binary file (.upb), and automatically adds them to the list of Files to be Downloaded.

5. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

Profibus Master configuration is now complete, but data exchange parameter configuration is required. For more details, see configure the Profibus Slave Input data and Output data.

20.11.5 Configure the Profibus Slave

The .gsd file provides detailed information about the Profibus Slave to the Profibus Master Configurator, including the modules that can be fitted. The Profibus Master Configurator is used to define the actual device configuration.

To configure the Profibus Slave:

1. Configure the Profibus Slave Properties. Edit these parameters to identify the operational details of the device.
2. Add a Profibus Slave Module. Profibus Slaves can be Compact devices or Modular stations. Modules are added to represent each additional module fitted to the device communicating via the Profibus network connected to the Profibus Master.

Note: Use Demand Data, if supported, to minimise the use of the communications bandwidth for parameters that require either occasional read or write access to data, e.g. autotune or a three term value, or complex read or writes that require a lot of data exchange, e.g. setting up and running a programmer.

3. Configure the Profibus Slave Module Properties. Edit these parameters to identify the module type at a defined position in the device.
4. Configure the Input data. This includes defining the address space for each required parameter in the device, and configuring the related parameters.
5. Configure the Output data. This includes defining the address space for each required parameter in the device, and configuring the related parameters.
6. Configure the Extended Diagnostics. This includes defining the address space for each particular parameter in the device, and configuring the related parameters.
7. Configure the Acyclic data. This includes adding individual records for each required parameter and configuring the related parameters.

Notes
1. Consult the device handbook for device parameter information.
2. Bits, Bytes and Words can be split or combined to provide sufficient address space for a device parameter, or to merge multiple device parameters to provide a single Digital Output that can be referenced in the LIN database.

20.11.6 Configure the Profibus Slave Properties

Configuration of the Profibus Slave Properties parameters consists of editing the module configuration, and configuring the General parameters and the User Parameter Data if supported by the device.

To configure the device Properties parameters,

1. Select a Profibus Slave to reveal the associated Properties page.

   Most device Properties page parameters are read only, but some can be edited by selecting a value from a Picklist, changing the module type or by changing the text value.
2. Edit the parameters on the Properties page, as required.

   - Configure the General parameters. This section refers to the device.
   - Configure the DP_Slave parameters. This section refers to the data exchange in the device.

   The parameters in this section are read only.

   - Configure the User Parameter Data parameters. This section refers to the module parameter configuration in the module.
3. Configure the Profibus Slave module. Compact or Modular stations are added one per slot, relating to the Profibus Slave device.

As each module type is assigned, a new Free Slot appears ready for configuring the next module type.

- Configure the module General parameters. This section refers to the module.
- Configure the Module parameters. This section refers to the data exchange in the module.

The parameters in this section are read only.

- Configure the User Parameter Data parameters. This section refers to the module parameter configuration in the module.

4. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each attempted operation and any warnings or errors that may have been detected. 'Save' also generates the GateWay file (.gwf), and Profibus Binary file (.upb) and automatically adds them to the list of Files to be Downloaded.

5. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

Profibus Slave configuration is now complete but data exchange parameter configuration is required. For details, see configure the Profibus Slave Data Exchange parameters.

20.11.7 Configure the Profibus Slave Module

Profibus Slaves can be Compact devices or Modular stations. Modules are added to represent each additional module fitted to the device communicating via the Profibus network connected to the Profibus Master.

Note: Some Module types must be allocated to specific Module locations. This may conflict with the .gsd file data, with the result that the Profibus Master Configurator will not allow the module to be configured in another position. To resolve this problem, the user must configure the position of the Module manually, if the .gsd file and Profibus Master Configurator allow it. Consult manufacturer's documentation for details.

To configure modules:

1. Expand the Profibus Slave that is to be configured. This is the device with modules that need to be configured.

Note: Use Demand Data, if supported, to minimise the use of the communications bandwidth for parameters that require either occasional read or write access to data, e.g. autotune or a three term value, or complex read or writes that require a lot of data exchange, e.g. setting up and running a programmer.

2. Select the next available free slot in the relevant Profibus Slave device. This reveals a **Module Unassigned** field on the Module Properties page to the right of the tree view.

Note: Demand Data will operate correctly only when configured to Module 1.

3. Select the **Module Unassigned** field and use the drop arrow to reveal a Picklist of Module types applicable to this device.

4. Select the appropriate Module type. This must correspond to the Module type that is fitted to this device.

   Once the Module type has been selected the tree view is automatically refreshed, showing the latest addition.

   - The Properties page of the Module type shows the parameters used to define this Module, providing access to the Module name field. This field may be edited to allow a more useful name to be applied to the Module. Some modules offer further parameter configuration, used to edit the details of a module.
   - The Inputs page and Outputs page show the total number of Bytes allowed in this Module.
   - The Split Rows button and Combine Rows button enable the configuration of the parameter address space. Bit, Byte, 16-bit Word and 32-bit Word Rows can be split or combined to provide adequate address space required for transferring data.

5. Insert or remove modules.

Additional modules can be added, using the available buttons across the top of the Profibus Master Configurator or the Context menu commands, to represent the modules that exist in the device.

6. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each operation it attempts and any warnings or errors that may have been detected. This also generates the GateWay file, .gwf, and Profibus Binary file, .upb, and automatically adds them to the list of Files to be Downloaded.

7. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

20.11.8 Configure the Profibus Slave Module Properties

Configuration of the Profibus Slave Module Properties parameters consists of configuring the General parameters and the User Parameter Data if supported by the device.

To configure the Module Properties parameters:

1. Select a Profibus Slave to reveal the associated Properties page.

   The parameters on this page identify the supported features of the device.

2. Select the Module to reveal the associated Properties page.

   The parameters on this page identify the supported features of the module.

3. Edit the parameters on the Properties page, as required

   - Configure the General parameters. This section refers to the device.

   All other sections are device dependent.

   - The Module parameters are read only, and refer to the data exchange in the module.
   - Configure the User Parameter Data parameters. This section refers to the module parameter configuration in the module.

4. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each operation it attempts and any warnings or errors that may have been detected. This also generates the GateWay file, .gwf, and Profibus Binary file, .upb, and automatically adds them to the list of Files to be Downloaded.

5. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

Profibus Slave Module configuration is now complete but data exchange parameter configuration is required. For details, see configure the Profibus Slave Input data and Output data.

20.11.9 Configure the Profibus Slave Input data and Output data

Configuring the device Input data defines the parameters that will be read from and written to the Profibus Slave by the Profibus Master. Parameters added to these pages automatically appear on the Total Inputs page and Total Outputs page of the Profibus Master.

The size of each device parameter depends on the specific Profibus Slave. The Profibus Master Configurator has been designed to allow Words, Bytes, and Bits to be split and/or combined, providing precise access to the address space for allocating device parameters.

Note: The tab at the top of the Inputs page and Outputs page shows the amount of address space used by the device. This value can be used to calculate the available address space.

To configure the Input data or Output data parameters,

Note: The following instructions apply equally to the configuration of Output data.

1. Expand the Profibus Slave and the required module that is to be configured from the Profibus Master Configurator tree view. This is the device and module with parameters that are to be read from or written to the Profibus Slave. It may show modules already configured for this device.

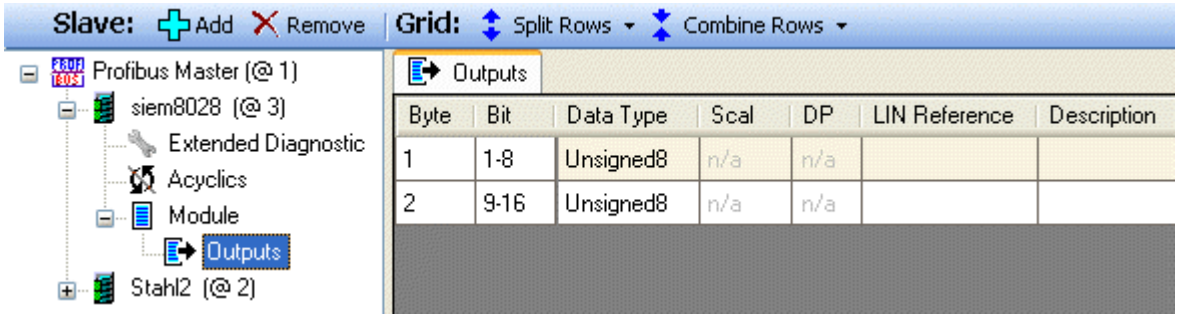
Note: Compact devices will show the pre-defined Module configuration. During device configuration there is no opportunity to change the module configuration.

2. Select the Inputs icon if configuring the Inputs, or Outputs icon if configuring the Outputs, from the Profibus Master Configurator tree view. This displays a table used to assign LIN database block fields to device parameters that are to be read from or written to the Profibus Slave, as appropriate.

   The Inputs page and Outputs page show the total number of Words, and corresponding Bytes and Bits available in the Module.

   i. Decide which parameter from the Profibus Slave is required. This will require information sourced from the manufacturers' documentation. The documentation should provide a list of parameters and the related address space required by the selected device parameter.

   ii. Configure the Input page or Output page rows to provide adequate address space for the required device parameter. Split or combine rows of Words, Bytes and Bits, to configure the required address space.



Note: Words, Bytes and Bits can also be split and combined using the context menu commands.

   iii. Configure the Data Type of the device parameter.
   Use the Data Type feature to configure the format of a returned value.

   iv. Configure the Scaling of the device parameter.
   Use the Scaling feature to define and scale Profibus and LIN range limits of a returned value.

   v. Configure the Decimal Place feature of the device parameter.
   Use the Decimal Place feature to configure the dividing factor for a returned value.

   vi. Associate a variable, database field, to the device parameter. This links the value from the device parameter to a specific field in the database.
   The associated variable is validated as it is inserted. A successfully validated variable is indicated by the ✓. An unsuccessfully validated variable is indicated by the ✗.

   vii. Add or edit the description. This is used to help explain the configuration of the Extended Diagnostics parameter.

3. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each operation it attempts and any warnings or errors that may have been detected. This also generates the GateWay file, .gwf, and Profibus Binary file, .upb, and automatically adds them to the list of Files to be Downloaded.
4. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

20.11.10 Configure the Profibus Slave Acyclic data

Note: Some instruments can operate only as a Profibus Slave when communicating via a Profibus network. Such instruments must be configured using ' Modbus Tools'.

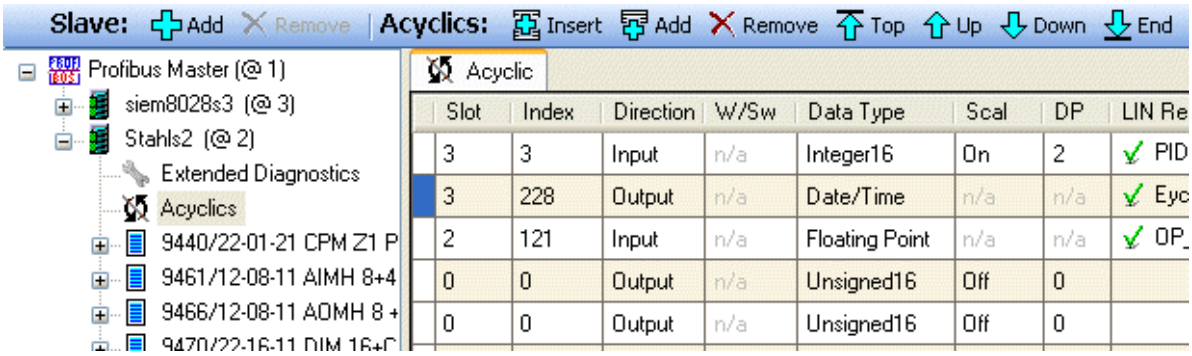
Input data and Output data is regarded as being assigned to Modules that can be addressed using Slot number and Index, as shown on the Acyclic page. The device parameters added to the Acyclic page are read from the Profibus slave or written to the Profibus Master in Slot number order, when available time (gap) remains in the programmed cycle. The gap time allows the Profibus Master to set up an acyclic connection to any device and exchange data.

Note: Acyclic data exchange continues until the remaining time has elapsed.

To configure an Acyclic data record,

1. Select the Acyclic icon from the Profibus Master Configurator tree view associated with the Profibus Slave. This shows a window that is used to list each device parameter that is to be read from or written to the Profibus Slave in parallel to the cyclic data.
   The Profibus Master Configurator reveals context related buttons used to add, remove, and order acyclic records, as appropriate.

2. Add an Acyclic record. Each record represents a device parameter at a defined Slot number (Module address), and Index (Word, Byte, or Bit address).



i. Click on the **Add record** button to create a record. Records are appended to the end of the list if the Add Record button is used, but can be ordered appropriately using the buttons available.

Note: The functions of the buttons shown above are also available via a context menu that appears when a Record in the Acyclic page is right-clicked on.

ii. Configure the Slot number and Index address. This is used to identify a value at a specific address space.

iii. Configure the data exchange direction. This is used to define the direction the data will be transferred.
   - o Select Output to write the data to the Profibus Slave.
   - o Select Input to read the data from the Profibus Slave.

iv. Configure the Data Type of the device parameter.
   Use the Data Type feature to define the format of the value.

v. Configure the Scaling of the device parameter.
   Use the Scaling options to define and scale Profibus and LIN range limits.

vi. Configure the Decimal Place feature of the device parameter.
   Use the Decimal Place feature to configure the dividing factor for a returned value.

vii. Associate a variable, database field, to the device parameter. This links the corresponding Profibus Data Type parameter to a specific field in the database.

Add or edit the description. This is used to help explain the configuration of the device parameter.

Note: Use this field as much as possible to help describe the Profibus data exchange configuration.

3. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each operation it attempts and any warnings or errors that may have been detected. This also generates the GateWay file, .gwf, and Profibus Binary file, .upb, and automatically adds them to the list of Files to be Downloaded.

4. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

20.11.11 Configure the Profibus Slave Extended Diagnostics

The configuration of Extended Diagnostics depends on the Profibus Slave type, because some devices do not support these additional related Alarm and Status messages.

To configure the Extended Diagnostics parameters,

1. Expand the Profibus Slave that is to be configured from the Profibus Master Configurator tree view. This is the device with additional Alarm and Status message parameters that are to be read from the Profibus Slave in the event of a device alarm or change of state.

2. Select the Extended Diagnostics icon from the Profibus Master Configurator tree view. This shows the list of Extended Diagnostics parameters that are supported by this device.

Note: A Description of each supported Extended Diagnostics parameter may already exist. This can be used to simplify the configuration of the block field in the database.

3. Configure the address space for the required device parameter.

Tip!
Combining individual Extended Diagnostics parameters can help reduce the number of required Digital channels, but additional configuration may be required to ensure correct operation.

i. Select the row corresponding to the Extended Diagnostics device parameter that is to be configured.
ii. Configure the rows to provide adequate Byte and Bits, address space, for the required device parameter.
iii. Associate a variable, database field, to the device parameter. This links the value from the device parameter to a specific field in the database.
The associated variable is validated as it is inserted. A successfully validated variable is indicated by the ✓. An unsuccessfully validated variable is indicated by the ✗.

4. Add or edit the description. This is used to help explain the configuration of the Extended Diagnostics parameter.

5. Save the Profibus configuration. Pressing the Save button launches the Build window displaying a record of each operation it attempts and any warnings or errors that may have been detected. This also generates the GateWay file, .gwf, and Profibus Binary file, .upb, and automatically adds them to the list of Files to be Downloaded.

6. Download the Instrument configuration to the LIN instrument operating as a Profibus Master.

20.11.12 Configure the Slot number and Index address

The Slot number and Index address of LIN instruments can determined using a calculation based on a Canonical Modbus address or actual Modbus address of a device to define a specific block of data.

Note: The Slot number and Index values of third-party products are determined using methods applicable to the device only.

To configure the Slot number and Index address,

1. Configure the Slot number. This is used to identify the Profibus Slave Module associated with an individual block of data that will be transferred.

2. Configure the Index address. This is used to identify an individual block of data assigned to a Module that will be transferred.

Note: Each individual block of data will not exceed 244 Bytes.

20.12 USE THE DATA RECORDING CONFIGURATION

The Data Recording Configurator is used to define the parameters to be recorded, and to configure their appearance in Review software. Parameters from the LIN database of an instrument are added to a Data Recording Group.

A data Recording icon appears in the Contents pane in LINtools when a database of an instrument that supports Data Recording is opened. It appears after the first parameter is specified and the required DataRec compound has been generated. When the Data Recording Configurator is started, pages are displayed that provide a means of visualising each parameter, assigning additional parameters to a group, and configuring the particular display of a selected parameter.

Notes:
1. Data Recording files will not be archived unless an FTP server has been configured. An FTP Server provides an archive destination for Data Recording (.uhh) files, produced by a specified instrument. The file can then be accessed by the Review software.
2. Configuring a computer to operate as an FTP Server should be attempted by an experienced computer engineer, and should comply with local company policies. This is beyond the remit of this document.

20.12.1 To use the Data Recording Configurator,

1. Configure the Data Recording configuration. This requires in-depth knowledge of the plant/system to ensure that all the relevant information is recorded and saved to the .uhh file. Parameters are defined by selecting the Add to Data Recording command from the context menu of a block or block field in LINtools.

Note: The Header block identifies the software version, that indicates the supported instrument functionality. If the Header block software version does not support the required functionality it must be changed when editing an existing database.

   i. Open the Data Recording Configurator. This is used to generate the Data Recording Configuration file, .uxg. This file must be downloaded to the instrument to ensure the correct Data Recording strategy is used.
   ii. Edit the Group configuration. This is used to control and manage Groups in the Data Recording Configuration. New Data Recording Groups can be added, and the Data Recording of existing groups can be enabled and disabled.
   iii. Edit the Block configuration. This provides information about the blocks selected for Data Recording. Further block Alarm fields can be selected, and added to a selected Data Recording Group, and included in the Data Recording Configuration.
   iv. Edit the Field configuration. This provides information about each block field selected for Data Recording. Further block fields can be added to a selected Data Recording Group, customised for display in Review, and included in the Data Recording Configuration. If this page shows only the Field Name and Group columns, **Show Visualisation Parameters for fields** on the **Options** page is ☐.

   Specific configuration of individual block fields can be enabled by setting the **Show Visualisation Parameters for fields** on the **Options** page ☑.

Note: The Options page contains editable parameters that control the Data Recording configuration.

   v. Edit the Option configuration. This provides parameters used to customise the block fields selected for Data Recording and that can be displayed in Review.

2. Save the Instrument configuration. This saves the Data Recording Configuration file, .uxg, and automatically adds them to the list of Files to be Downloaded.

3. Download the Instrument configuration to the LIN instrument. Pressing the Download button connects to the instrument and copies the specified files.

   The Instrument configuration folder contains all the files used by the Instrument to perform its required function. To ensure successful data recording and archiving

   Upgrade the Licence. Data Recording is Licence protected, and requires a D10 to D90 Licence that supports function block codes K to T. Use the Licence Tool from the Tools menu to request a Licence upgrade.

Note: This is not applicable if a Licence was ordered when the instrument was purchased.

- Define the FTP Server. This ensures that all .uhh files will be saved to a secure environment.

Note: The Store and Forward tool can be used to replace periods of lost data in InSQL and the Alarm databases with recorded data from the .uhh files in LIN instruments and 5000/6000 recorders.

To display data recording files

- Configure Review software. This software can be configured to gather the .uhh files on the FTP Server at a determined period or on-demand.

Note: Ensure Review software (Review), is installed on the computer used to access the .uhh files gathered from the FTP Server. Review is supplied on a separate CD.

20.12.2 Configure Data Recording

This is the process of configuring individual block fields selected for Data Recording to change the:

- block, field and/or group configuration
- display in the Review software

To configure Data Recording

1. Define the parameters for Data Recording. Add block fields to data recording groups.
2. Configure the block, field and/or group, and visualisation. When the required block fields have been defined, individual block fields can be configured.

i. Open the Data Recording Configurator. This is used to generate the Data Recording Configuration file, .uxg. This file must be downloaded to the instrument to ensure the correct Data Recording strategy is used.
ii. Edit the Group configuration. This is used to control and manage Groups in the Data Recording Configuration.
iii. Edit the Block configuration. This provides information about the blocks selected for Data Recording.
iv. Edit the Field configuration. This provides information about each block field selected for Data Recording. If this page shows only the **Field Name** and **Group** columns, **Show Visualisation Parameters for fields** on the **Options** page is  .

   Specific configuration of individual block fields can be enabled by setting the **Show Visualisation Parameters for fields** on the **Options** page  .

Note: The Options page contains editable parameters that control the Data Recording configuration.

v. Edit the Option configuration. This provides parameters used to customise the block fields selected for Data Recording and that can be displayed in the Review software.

To ensure successful data recording and archiving

- Define the FTP Server. This ensures that all .uhh files will be saved to a secure environment.
- Upgrade the Licence to provide data recording support for the configured instrument. Data Recording is Licence protected. Data Recording requires a D10 to D90 Licence, and supports function block codes K to T.

  To update the Licence launch the **Tactician Licence Tool** from the Tools menu.

Note: This is not applicable if a Licence was ordered when the instrument was purchased.

20.12.3 Define the parameters for Data Recording

Specifying parameters for recording is achieved by selecting the required parameters, then using the Add to Data Recording command from the context menu in LINtools.

Note: A  indicates that the parameter has been selected for Data Recording.

To define the parameters for Data Recording

1. Locate and select the block containing the parameter that is required for Data Recording, to display the block field in the Object Properties Pane.

   A context menu including the Add to Data Recording command is displayed when a Block, Field, Compound, or multiple Blocks are selected. If a Block, Compound, or multiple Blocks are added to the data recording, a default selection of fields is automatically added to the Data Recording configuration.

Note: When using a layered database, block fields can be added or removed from any layer.

2. Select the required parameter to reveal the context menu and select the **Add to Data Recording** command.

   If a Group does not exist, select the New Group command to display the **Add New Group** dialog.

   If a Group does exist, simply select the required group.

Notes:
1. This Group name appears in the *Name* field of the associated RGROUP, and is used as a reference in the Review software.

2. Use the Find utility to locate and select multiple block fields for Data Recording. The Add to Data Recording command is available from the context menu.

3. When this command is completed a **DataRec** compound is automatically created.

   This compound contains a RGROUP block, a RARCDIAG block, and a RMEMDIAG block. As further Data Recording Groups are added, a single RGROUP block is automatically created for each new group.

Note: Parameters can also be added to a specified group using the Block Name field on the Blocks page, or the Field Name field on the Field page available in the Data Recording Configurator.

Additional commands available on the context menu provide the option to

- go to the RGROUP block that controls the selected Data Recording Group
- Remove the selected block field, without deleting the block, from the defined Data Recording Group

20.12.4 Edit the Group configuration

This provides a tabular way to add, delete, enable and disable individual Data Recording Groups in Data Recording Configuration.

Notes:
A maximum of eight or 16 groups can be configured according to model, with each group supporting a maximum of 128 data record points. If MinMax on the Fields page is On, the associated value accounts for two data record points in the Review software.
2. When using a layered database, Groups can only be added from the base layer.

To edit the Group configuration:

1. Select the **Group Name** field, and enter an appropriate name for the group, i.e. group1. The **Group Name** is a maximum of 20 characters. When selected the associated RGROUP block appears in the Object Properties pane.

Note: Any changes to time period between recording values, UpdateA or UpdateB, if SelectB is TRUE, automatically changes the Est. Memory Duration.

2. Consider **Enable Recording** for the group.
   - If **Enabled**, the parameters assigned to this group will be recorded
   - If **Disabled**, the parameters assigned to this group will not be recorded
3. Observe the **Est. Memory Duration** field. This provides an estimate of the Data Recording memory available in the instrument. This calculation is based on the number of parameters in the group, and the number of groups in the Data Recording strategy.

Note: The RMEMDIAG block will assist with diagnosing flash memory faults when recording values from selected parameters, and the RARCDIAG block will help diagnose faults when archiving .uhh files from the instrument to the specified FTP Servers.

Additional commands available on the context menu provide the option to

- go to the RGROUP block that controls the selected Data Recording Group
- Delete the selected Data Recording Group

20.12.5 Edit the Block configuration

Note: The RMEMDIAG block will assist with diagnosing flash memory faults when recording values from selected parameters, and the RARCDIAG block will help diagnose faults when archiving .uhh files from the instrument to the specified FTP Servers.

This provides a tabular way to add, delete, and manage blocks assigned to Data Recording Groups in the Data Recording Configuration. To configure the Blocks:

1. Select the **Block Name** field, and select the required block from the Picklist displayed. Adds a default selection of the block's fields to the data recording. When the required block is selected, the **Recorded Alarms** shows **All**. This indicates that all Alarms for this block will be recorded. **Recorded Alarms** is set to **All** by default, but the recording of specific alarms can be configured by editing the **Recorded Alarms** field.
2. Select the relevant **Group** field, and select the required **Group Name** from the Picklist displayed. This is used to assign the block, identified in the **Block Name** field, to the Data Recording group, identified in the **Group** field.
3. Select the relevant **Recorded Alarms** to reveal the **Select alarms to record** dialog. This dialog provides the selection of individual Alarm parameters from the selected block.
   - Set **Record All**  to indicate that all Alarms will be recorded.
   - Set individual Alarm parameters  to indicate that only the defined Alarms recorded.

   This column shows the Alarm parameters that will be recorded for this block.

Additional commands available on the context menu provide the option to:

- go to the specified block in the database
- remove the defined parameter from the specified Data Recording Group

20.12.6 Edit the Field configuration

This provides a tabular way to add, delete, and manage block fields assigned to Data Recording Groups in the Data Recording Configuration.

Note: The RMEMDIAG block will assist with diagnosing flash memory faults when recording values from selected parameters, and the RARCDIAG block will help diagnose faults when archiving .uhh files from the instrument to the specified FTP Servers.

To configure the Fields

1. Select the **Field Name** field, expand the required block and select the field from the Picklist displayed. This is used to specify the parameter selected for Data Recording.

When the required field is selected, it appears in the **Field Name** column, with a default configuration. This indicates that this field will be recorded.

2. Select the relevant **Group** field, and select the required **Group Name** from the Picklist displayed. This is used to assign the field, identified in the **Field Name** field, to this Data Recording group.

Note: Further configuration of a selected Field is optional and only available if the Show Visualisation Parameters for fields on the Options page is  . The configuration of the following parameters apply to the display of a Trend when shown in the visualisation software, Review.

3. Select the relevant **Description** field, and enter a unique description for the field using up to 16 characters. This description is shown in the Review software, and is used to identify the associated block.field value.

4. Select the relevant **Format** field, and, if required, select a format from the Picklist displayed. This indicates the format of the defined value when displayed in the Review software. The format selection is dependant on the block.field type.

5. Select the relevant **Colour** field, and, if required, change the default colour. This is the colour of the defined value when displayed in the Review software.

6. Select the relevant **MinMax** field, and enable (On) or disable (Off) the adaptive recording functionality.

Note: When MinMax is On, the associated value accounts for two trends in 'Review' software.

7. Configure the **SpanLow** and **SpanHigh** fields. These are used to define the limits of the zone shown in the visualisation software. **SpanLow** values will be displayed closer to ZoneLow, and **SpanHigh** values will be displayed closer to ZoneHigh.

Note: If 'inverted zero' is required then set SpanLow greater than SpanHigh.

8. Configure the **ZoneLow** and **ZoneHigh** fields. This is used to define the area of the chart showing the defined value when displayed in 'Review' software, e.g. settings of 50 and 100 cause the trace to appear in the top half of the chart.

9. Configure the **Active** and **Inactive** fields. This is used to define a text string using up to 16 characters to represent the digital zero (0) or non-zero (1) recorded value.

   i. Select the relevant **Active** field, and, if required, change the default text, TRUE, to a description that represents the recorded non-zero value.
   ii. Select the relevant **Inactive** field, and, if required, change the default text, FALSE, to a description that represents the recorded zero value.

Additional commands available on the context menu provide the option to

- go to the specified field in the database
- Remove the defined parameter from the specified Data Recording Group

20.12.7 Edit the Option configuration

This provides a selection of parameters used to allow customisation of individual values in the Data Recording Configuration.

To configure the Options

1. Configure the **Show Visualisation Parameters for fields**. This is used to display additional parameters that provide control of visualisation parameters that appear on the **Fields** page.

   - Set **Show Visualisation Parameters for fields**  to allow individual customisation of a defined parameter selected for data recording.
   - Set **Show Visualisation Parameters for fields**  to prevent individual customisation of the defined parameter selected for data recording.

2. Configure the **Instrument name area** fields. These fields provide options used to identify the Data Recording Groups configured in this instrument.

   - When **Use default name for instrument area** is selected, the data recording group name shown in the Name field in the RGROUP block remains as the name entered when the data recording group was created.
   - When **Use custom name** is selected, the associated entry preceeds the data recording group name shown in the, Name field in the RGROUP block, and the name entered when the data recording group was created. This provides a greater level of identification of the parameters selected for data recording when using a Review software.

20.12.8 Define the FTP Server

Defining the required FTP servers provides the destination of .uhh files in the instrument. Configuring the Archiving category in the Instrument Properties allows the .uhh files to be pushed to all configured FTP Servers.

Each required FTP server is specified via the Archiving section of the Instrument Options page in the Instrument Properties dialog.

To configure the FTP servers,

1. Select the LIN Instrument Folder in the LINtools Contents pane to show the context menu.
2. Select the **Properties** command to launch the Instrument Properties dialog.
3. Select the Instrument Options page to show the hardware configuration of the instrument. The Instrument Option page shows configuration data for various aspects of the instrument, e.g. HMI settings, Modbus or Profibus ports configuration, Archiving, etc..
4. Select the **Archiving** icon to reveal parameters that are used to configure the archiving of .uhh files.
5. Select the **Intervals** field and enter or use the spin button to define the total number of minutes between transfers of .uhh files to the FTP server.
6. Define the **FTP servers**. Up to 3 FTP servers can be configured to provide adequate backup when archiving .uhh files.

   1. Decide which FTP server will be configured as **FTP Server 1**, **FTP Server 2**, and/or **FTP Server 3**, as indicated by related FTP Server titled field.

Note: One or more FTP Servers can be configured, but a minimum of 2 must be configured to provide redundant operation.

   2. Set **Enabled** field TRUE. This enables communication between this instrument and the related FTP **S**erver.
   3. Enter the archive **Host IP** address of the related FTP Server. This identifies a specific Computer or instrument in the network.

Note: Not all instruments can be configured to operate as an FTP server, see specific instrument documentation.

   4. Enter the destination **Directory** path in the related FTP Server. This defines the directory (folder) on the FTP Server where the .uhh files are to be stored.

Note: For security reasons FTP servers normally only permit access to a limited area of their host computer by re-mapping the directories. It is the re-mapped name that must be entered here.

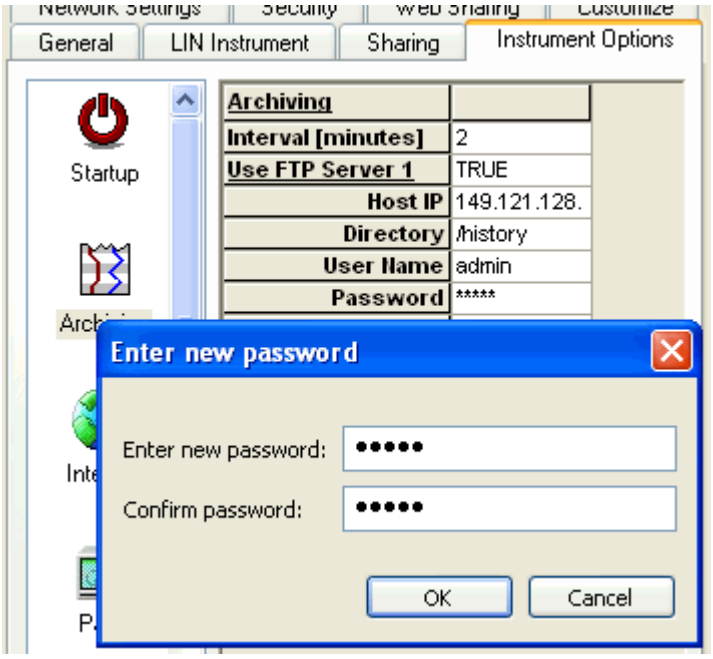
   5. Enter the **User Name**, and corresponding **Password**. This is used to provide a level of security to prevent unauthorised write access to the specified FTP Server.

      A Password dialog appears when the **Password** is selected. This Password dialog provides User name confirmation.

      - Enter and confirm the Password used to access the archive directory on the FTP Server defined by the Host IP.

Note: If an incorrect User Name or Password is entered, relevant error fields in the RARCDIAG block will be set TRUE when attempting to archive the .uhh files.

An example of Archiving Configuration



20.12.9 Configure Review

Review software allows the display and printing of archived Data Recording (.uhh) files, from suitable recorders, data acquisition units, etc.. The .uhh files are transferred from the remote instrument to a database on the computer. The database stores the recorded data by instrument tag, group name and point identifier.

Notes:

1. Recorded data from multiple archive files from the same group and the same instrument are stored together on the database, so that data can be viewed across files.

2. Refer to the Review help file for full details.

To configure Review software

1. Install Review using the appropriate CD.

Remember
The computer will require rebooting when Review has finished installing.

2. Launch **Review**. Further configuration is required to define the relationship between the FTP Server and the Review software, and to display the recorded values in the .uhh files.
   The Review software can be launched from

Start > **Programs** > **…** > **Review**

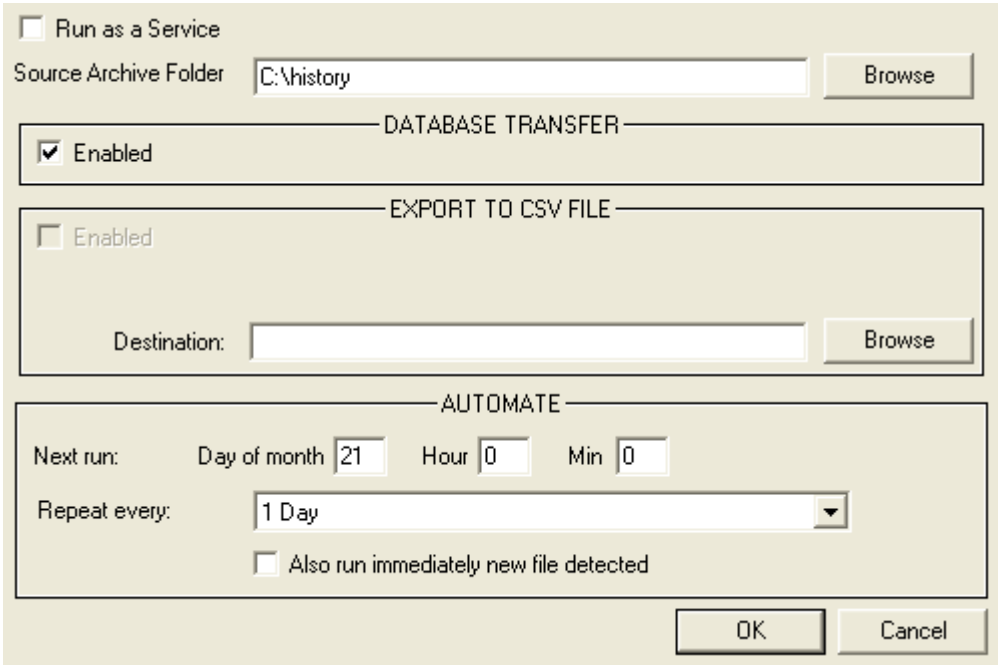
Note: '…' denotes the default path.

   If a Master Archive Folder has not been configured already a prompt appears. Select **Yes** to confirm that a Master Archive Folder is required. Select **OK** to confirm that the Master Archive folder is required at the location defined in the Master Archive Folder field.

3. Configure Data Transfer requirements. This defines when the .uhh files are transferred from the identified FTP Server to the Review database.

Note: If Security Manager has been installed additional security parameters must be configured and a Login and Password may be required.

i. Select **File**, **Transfer**, then **Scheduled Transfer** to display the **Scheduled Transfer from Folders** dialog.
ii. Define the Source Archive Folder. This is the folder on the FTP Server that retains the .uhh files.
iii. Select **Enabled**. This is used to specify that Data Transfer is allowed.

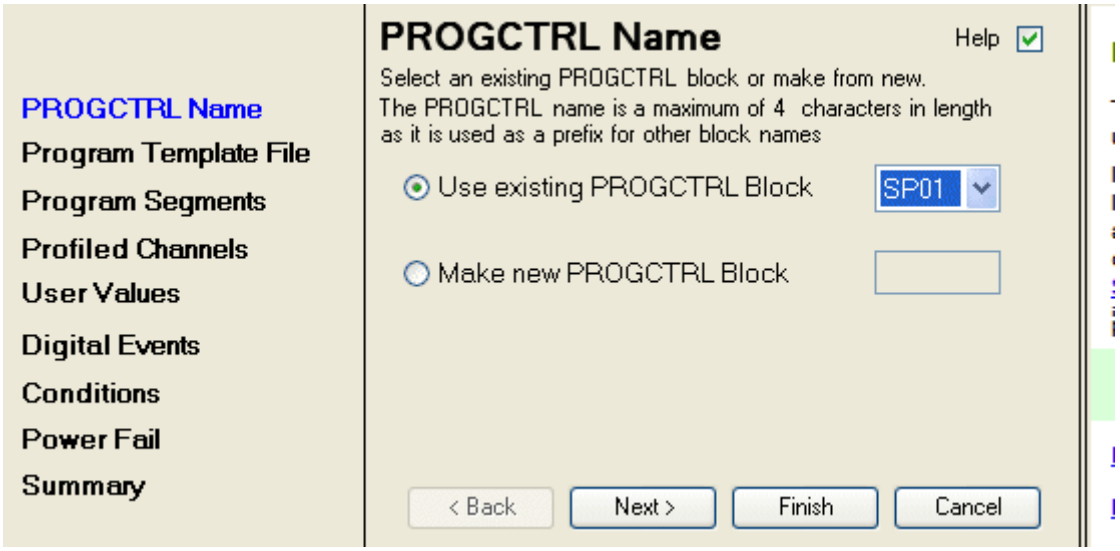


4. Create or open a chart. This is used to display the .uhh files in a chart form.

20.13 USE THE PROGRAMMER WIZARD

The Programmer Wizard is used to generate or edit a Program Template file and create a PROG_WIZ compound in the database. The wizard defines individual channels that can be profiled alongside digital events and user values and the maximum number of segments available to the Program.

Note: The Programmer Wizard creates blocks that can only control Programs produced using the Programmer Editor.



Note: Context help is displayed when the 'Help' checkbox is set .

To generate or edit a Program Template file, .uyw,

1. Launch the Programmer Wizard. Select **Tools** > **Programmer Wizard**.
2. Progress through the wizard until all appropriate fields are complete, The Finish button appears on the last page.
3. Click on the 'Finish' button to finalise the operation, and automatically generate the PROG_WIZ compound.

The PROG_WIZ compound contains, 1 PROGCTRL block, up to 8 PROGCHAN blocks one for each profiled channel, and up to 8 SEGMENT blocks per channel, providing a maximum of 32 program segments (each SEGMENT block offering 4 program segments).

It automatically lays the blocks out in a logical order. Any changes to this default layout are cancelled when the compound is closed.

20.14 CONFIGURE ALARM SUPPRESSION

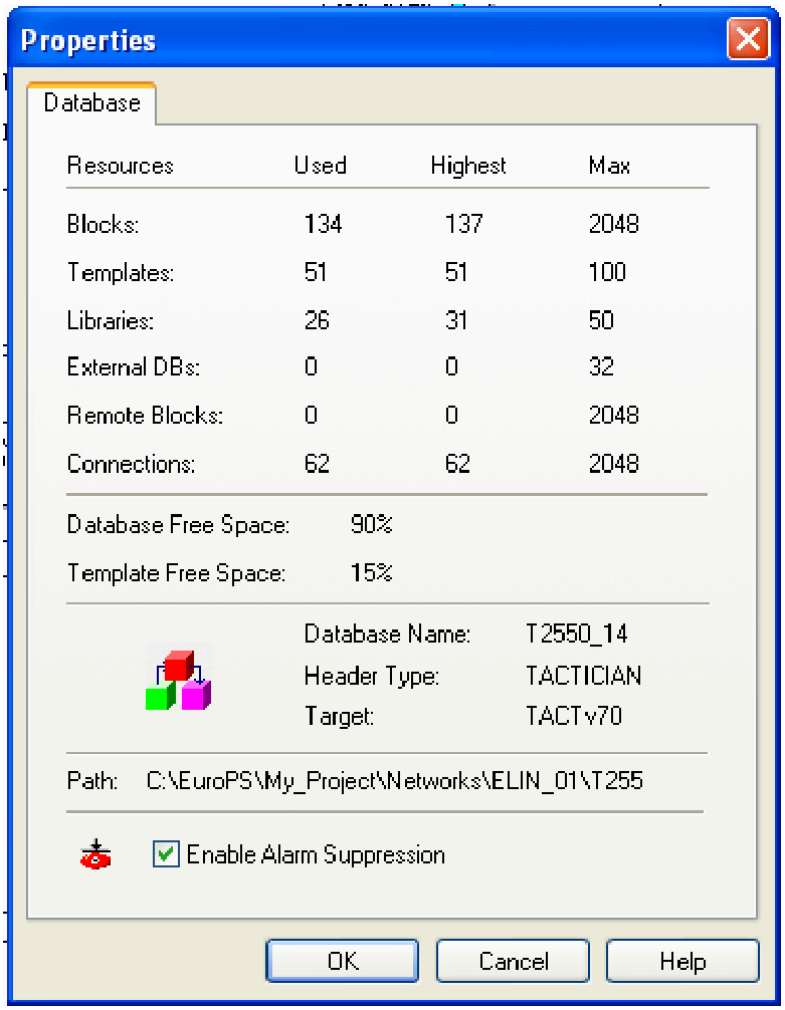
20.14.1 Enable Alarm Suppression

To enable Alarm Suppression functionality, select the ‘Loaded File Properties’ item from the LINtools ‘File’ menu to open the database properties display (shown below). ‘Tick’ the ‘Enable Alarm Suppression’ box by clicking within it.

The icon () denoting that Alarm Suppression has been enabled is also shown in the bottom right hand corner of the LINtools window, unless on-line connection mode or on-line configuration mode (not all instruments) is being used.

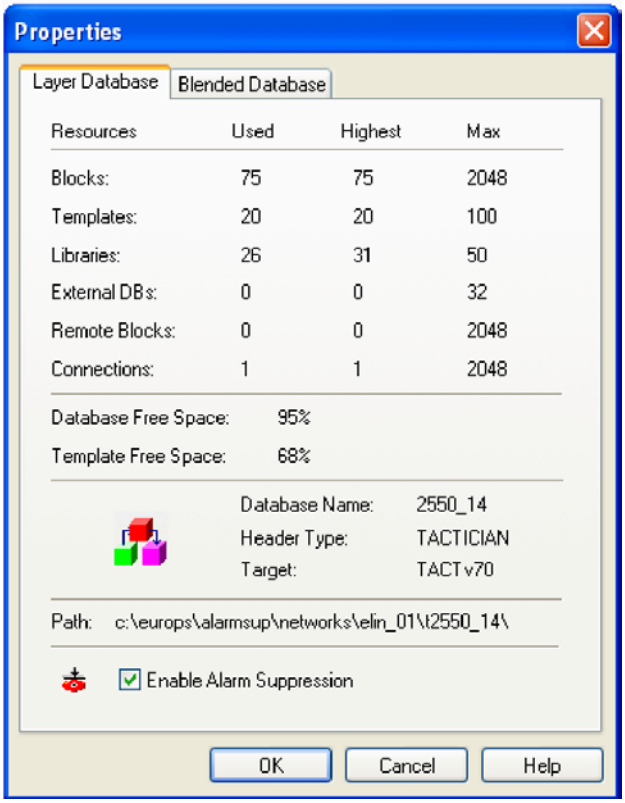
The enable checkbox and associated wording does not appear for databases created pre Alarm Suppression. In such cases, the database must be upgraded before Alarm Suppression can be enabled (see the 'Alarm Suppression User Guide', HA030272 for details).

Note: Enabling Alarm Suppression impacts on database size. Refer to the 'Alarm Suppression User Guide', HA030272 for further details

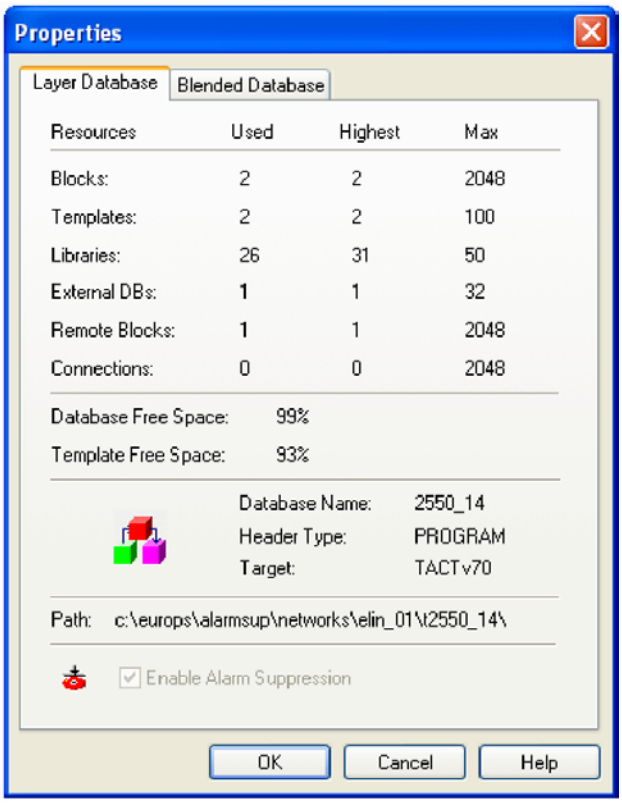

Properties
Database
Resources
Used
Highest
Max
Blocks:
134
137
2048
Templates:
51
51
100
Libraries:
26
31
50
External DBs:
0
0
32
Remote Blocks:
0
0
2048
Connections:
62
62
2048
Database Free Space:
90%
Template Free Space:
15%
Database Name:
T2550_14
Header Type:
TACTICIAN
Target:
TACTv70
Path:
C:\EuroPS\My_Project\Networks\ELIN_01\T255
Enable Alarm Suppression
OK
Cancel
Help

20.14.2 Layered and blended databases

For ‘Layered’ and ‘Blended’ databases, enabling Alarm Suppression from the ‘Loaded File Properties’ display as shown below can be performed only when the ‘base’ layer is the current database loaded into the LINtools Function Block Diagram (FBD) configurator.



When other associated layers are loaded into the LINtools FBD configurator it is not possible to enable or disable Alarm Suppression as the ‘tick box’ is ‘greyed out’ as shown below and is always the case when viewing the ‘Blended Database’ tab.



20.14.3 Alarm Suppression and SFCs

This is the primary method for applying Alarm Suppression configurations and strategies. This section covers only the additional ST attributes supported within SFCs to apply Alarm Suppression.

EXAMPLE SFC STEPS AND ASSOCIATED ST

The example below is based on two function blocks and an SFC(TC_Break) and implements a strategy where, in the normal process operating mode, all alarms in the PID block tagged 'TIC-001' are unsuppressed, but when the process is being steam cleaned, PID(TIC-001) alarms are suppressed as follows:

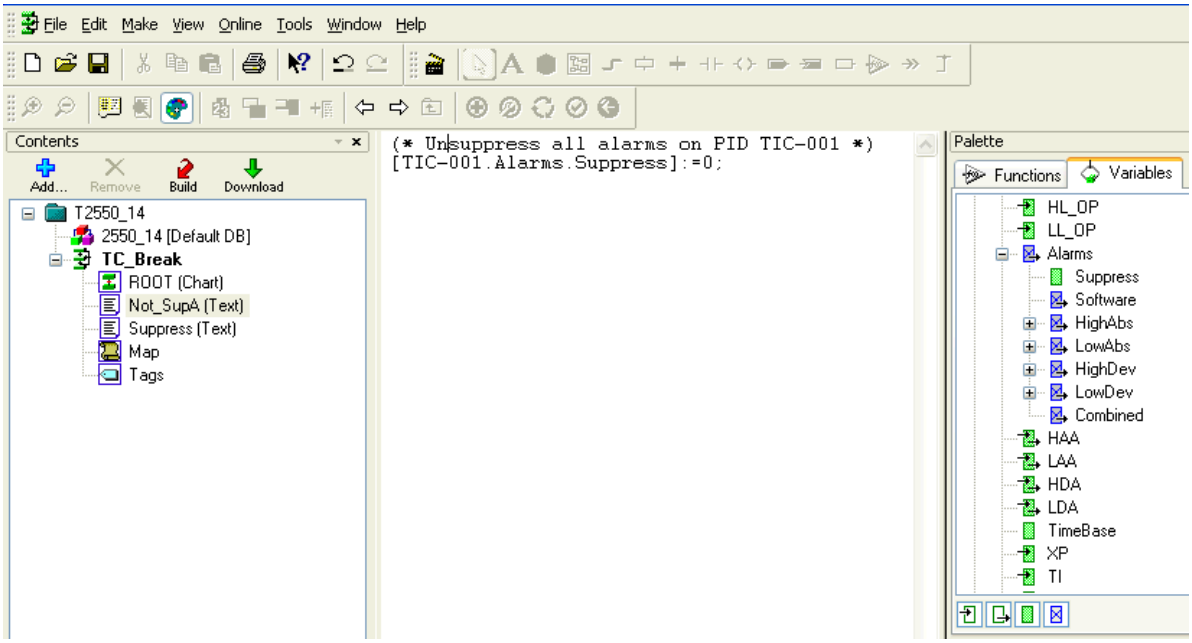
Absolute High Alarm - Indefinite

High Deviation Alarm - Indefinite

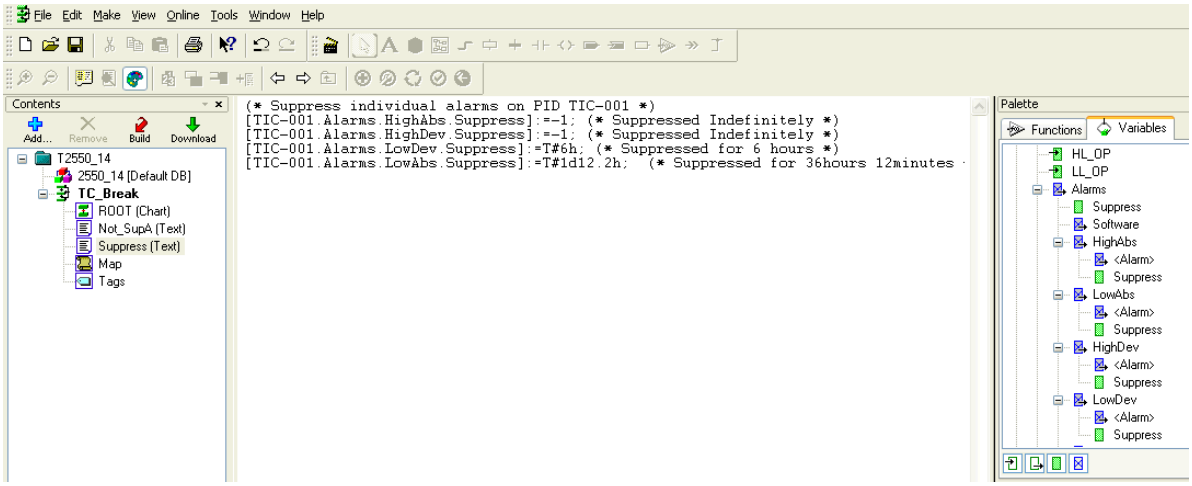
Low Deviation Alarm - 6hours

Absolute Low Alarm - 36hours and 12 minutes

Noting the Picklists showing the Alarm Suppress attributes, the ST for each of the two SFC 'Actions', 'Not_SupA' and 'Suppress' are shown respectively as below:



Example of Unsuppressing all block alarms



Example of Suppressing individual alarms

20.14.4 Alarm Suppression and Ladder Diagrams

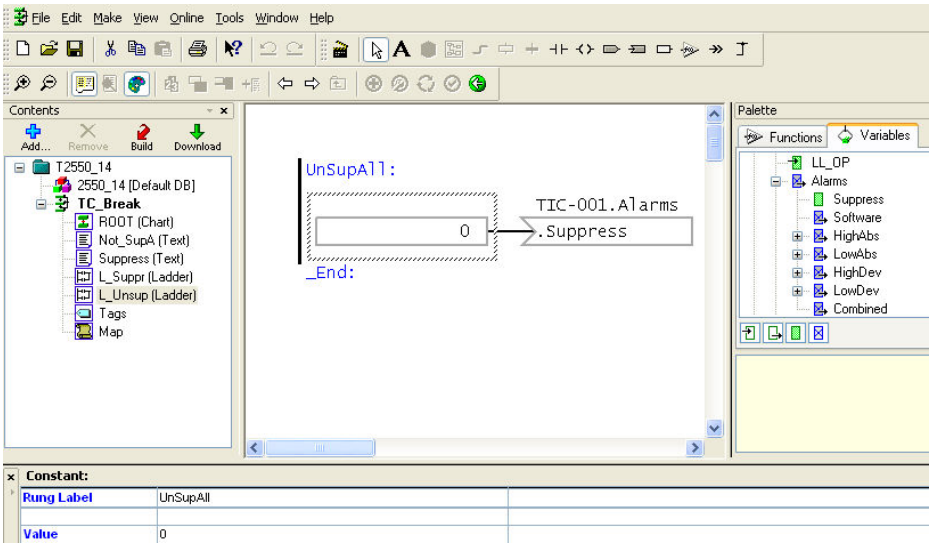
The same Picklist attributes are available for developing a Ladder diagram in the same manner as for creating an SFC. Using the example below and noting the Palette Picklists showing the Alarm Suppress attributes, the Ladder diagram for each of the conditions, 'L_Suppr' and 'L_Unsup' are shown respectively as follows:

Example Ladder Rungs and associated ST

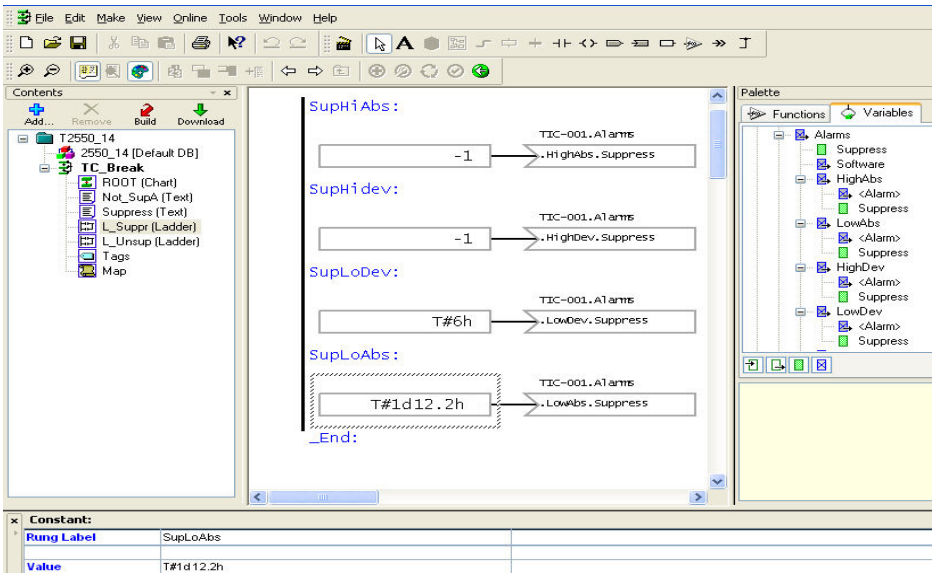
The example below is based on two function blocks and implements a strategy that in the normal process operating mode, all alarms in the PID block tagged 'TIC-001' are unsuppressed, and when the process is being steam cleaned, PID(TIC-001) alarms are suppressed as follows:

- Absolute High Alarm - Indefinite
- High Deviation Alarm - Indefinite
- Low Deviation Alarm - 6hours

- Absolute Low Alarm - 36hours and 12 minutes



Ladder Diagram for unsuppressing all alarms in 'TIC-001'

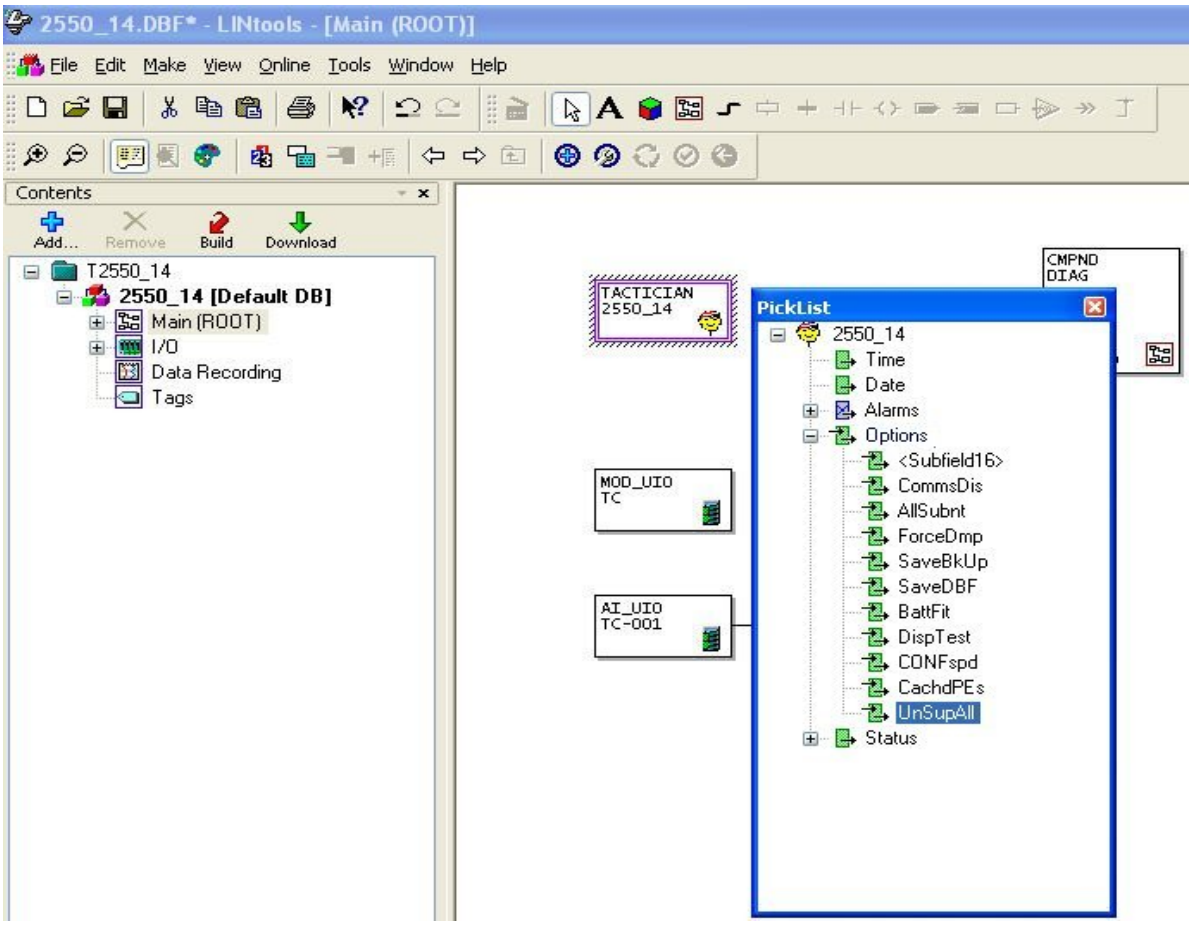


Ladder Diagram for suppressing individual alarms within 'TIC-001'

20.14.5 Header Block Unsuppression

Referring to the figure below, it is possible to wire to the 'UnSupAll' bit within the 'Options' field as shown in the 'Picklist'. When set TRUE it unsuppresses all alarms in that instrument's local database, regardless of whether they have been set to indefinite or with a time duration. The 'UnSupAll' bit then auto-resets to FALSE, i.e. the 'UnSupAll' bit reacts only to a rising edge. Wiring out from the 'UnSupAll' bit serves no purpose.

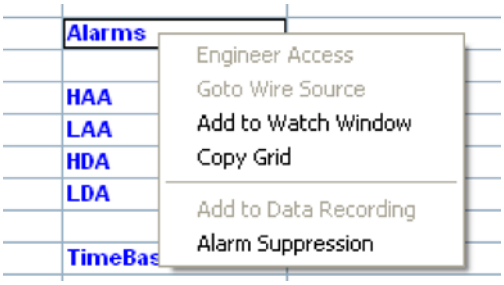
Note: The 'UnSupAll' bit reacts only to a rising edge. If held TRUE, Alarm Suppression continues to function as normal until the next rising edge



20.14.6 Alarm Suppression and On-line Connect

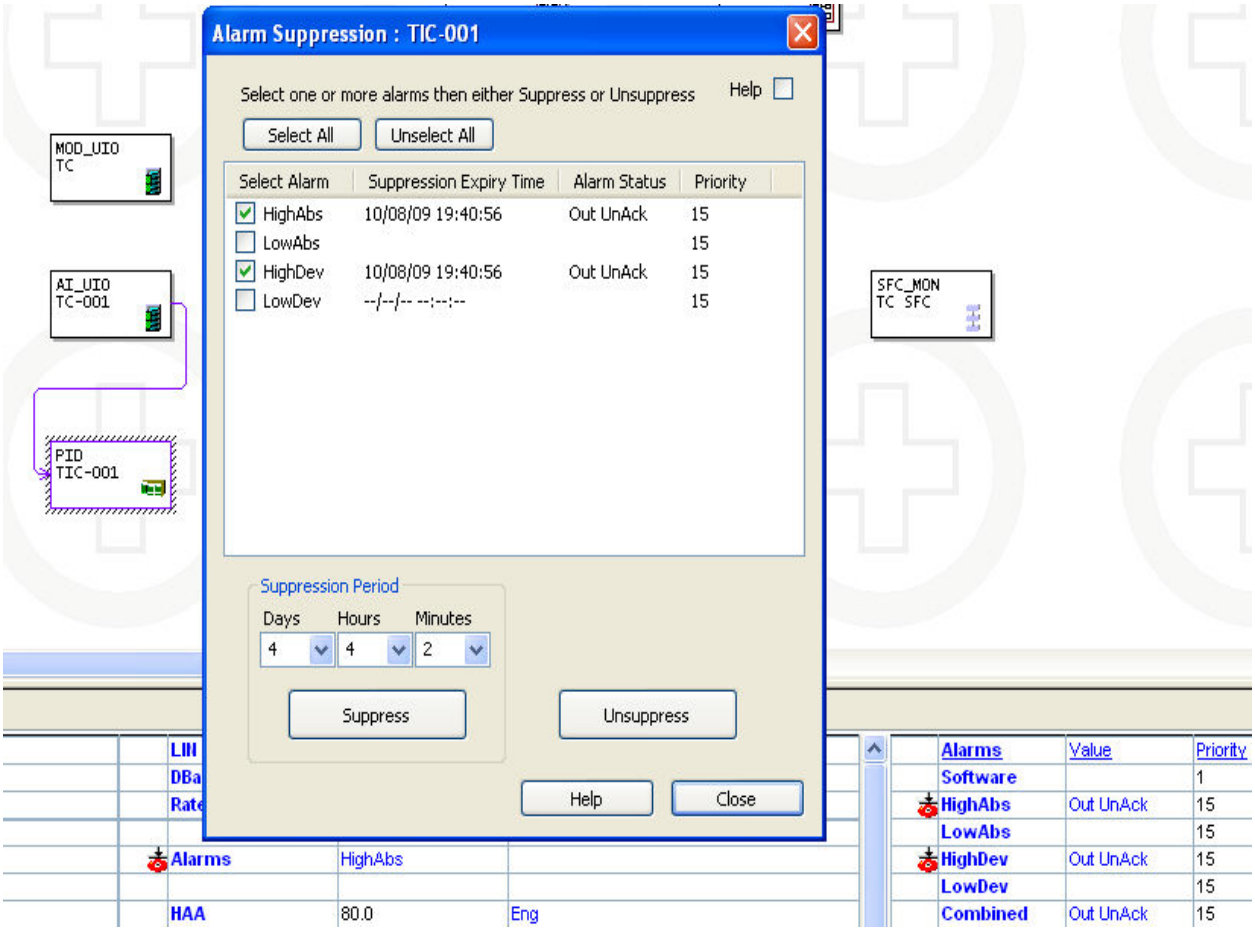
When using the online connection or on-line configuration mode (not available for all instruments) within LINtools, a facility for managing Alarm Suppression of each block is available to the user as shown in the figure below. Alarms can be suppressed regardless of whether they are active or inactive. Setting indefinite Alarm Suppression from this facility is prohibited.

To access the on-line Alarm Suppression facility, select the required block by single or double clicking within it, locate the 'Alarms' field and place the mouse cursor within it and then 'right click'. From the drop down menu shown below, select the 'Alarm Suppression' item.



Using the example as shown below, the Absolute High Alarm(HighAbs) and High Deviation Alarm(HighDev) have been suppressed for the time period shown in the 'Suppression Period' pane. This was achieved by selecting both alarms by 'ticking' the associated boxes, setting the Alarm 'Suppression Period' and finally clicking on the 'Suppress' button. The icon denoting alarms in suppression is shown adjacent to each current suppressed alarm and also the 'Alarms' field. Both alarms are in an 'Unacknowledged' state (section 2.4 provides further information). The Low Deviation Alarm(LowDev) is suppressed indefinitely, achieved via an SFC and denoted by '--/--/-- --:--:--'.

Note: Alarm Suppression only applies to Alarms set with a priority set between 1-15.



On-line Alarm Suppression facility

To unsuppress alarms, select the required alarms by 'ticking' the associated boxes and clicking on the 'Unsuppress' button. By ticking the 'Help' box, a help dialog pane opens giving further details.

HEADER BLOCK - SAVING CURRENT DATABASE

The database header block contains a facility to save the current database, normally located in the 'Options' menu. It should be noted that when using this facility the current Alarm Suppression timers are not saved.

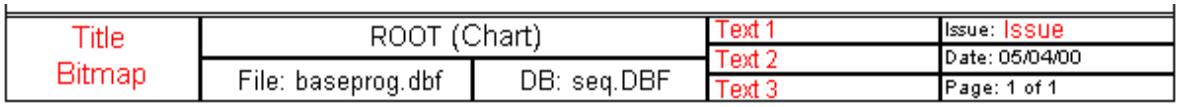
20.14.7 Alarm Suppression and ACTION Blocks

ACTION blocks are prohibited from supporting Alarm Suppression in the context of ST written within them. Alarm suppression can, however, be applied to 'ACTION' block alarms via, for example an SFC, i.e. An 'ACTION' block can have its alarms suppressed but cannot suppress them itself. This also applies to the RAW_COM block which is capable of supporting an associated ACTION.

20.15 CUSTOMISE LINTOOLS

20.15.1 Customise LINtools Printed Page Setup

Use the Page Setup dialog to configure the title block that appears at the foot of printouts produced via the **Print…** command. In the image below - the legends shown in red are the ones you can configure via the corresponding fields in the dialog:



To access the Page Setup dialog,

1. In the window **File** menu, select **Page Setup…**.

The dialog fields are:

Text 1, Text 2, Text 3

Free-format text strings. The image above shows where each string appears in the title block.

Issue

This string appears in the **Issue** cell, after 'Issue:' (see image above)

Title (Override)

Free-format text string appearing in the leftmost cell.

Bitmap path and filename

Enter the full path and filename of a Bitmap (*.bmp file). It will appear in the leftmost cell below the title (see image above).

Printed Border Layout

This image shows where each string appears in the title block.

20.15.2 Print LINtools configurations

Two kinds of printout of your LINtools configurations can be obtained:

- Page-based printouts with each compound, chart, ladder, ST action, etc., on a separate page, as listed in the **Contents** pane. Pages may include graphics, and every page has a set of customised title blocks at its foot. You can opt to print selected pages or page-ranges.
- An exportable single text-only file of your configuration.

20.15.3 To print a LINtools configuration in page-based format:

1. Set up a customised title block for the hardcopy, using the Page Setup dialog.
2. Click the Print toolbutton, or select **File** > **Print...** from the menu.
3. Specify the printing parameters in the **Print** dialog, then click OK.

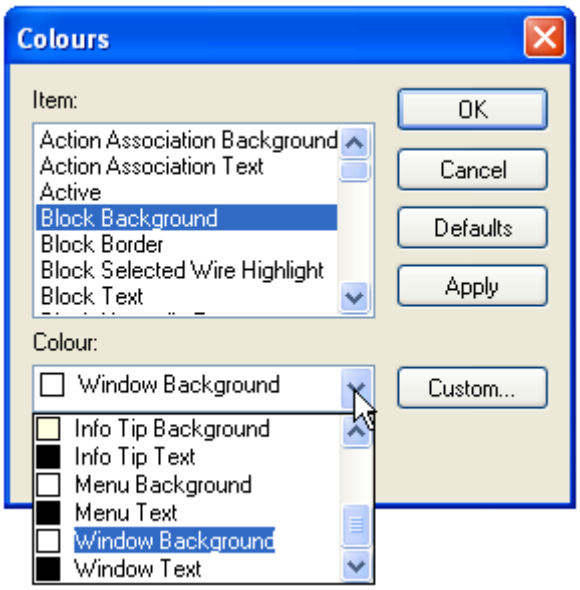
Note: It is worth looking at a preview of the printout first - click File > Print Preview.

TO GENERATE A LINTOOLS CONFIGURATION TEXT-ONLY FILE:

1. In the **File** pulldown menu select **Save Special** > **Text Listing**. A **Save As** dialog appears.
2. Edit the default **File name** and the **Save in** destination, if required, then click the **Save** button. The configuration is saved as a **text file**.
   - The text file layout corresponds (approximately) to the **IEC1131-3** standard.

20.15.4 Customise LINtools colour coding

LINtools permits various items' colours to be edited.



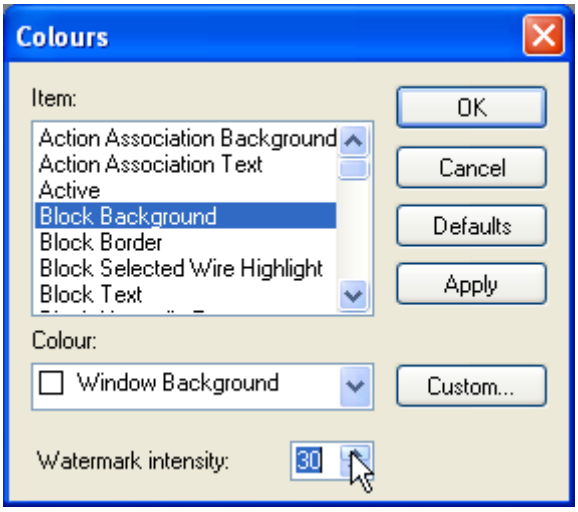
TO EDIT COLOUR CODING PROPERTIES

1. In the View > Options menu, click the 'Colours...' option to open the Colours display.
2. To change a default colour adopted by LINtools item, locate and click the required item and select a colour, either from one of the defaults in the pull-down menu (shown above) or via the 'Custom...' button (see below).
3. Click 'OK'. The new colour shows in the Colour window.
4. Finally, click 'OK' again to apply the customised colour scheme.

Notes

1. The Defaults button restores all the items to the default colour scheme.
2. Default colour red is adopted by 'Active' Ladder diagram elements, default colour black is adopted by 'Inactive' Ladder diagram elements.

TO EDIT WATERMARK INTENSITY SETTINGS

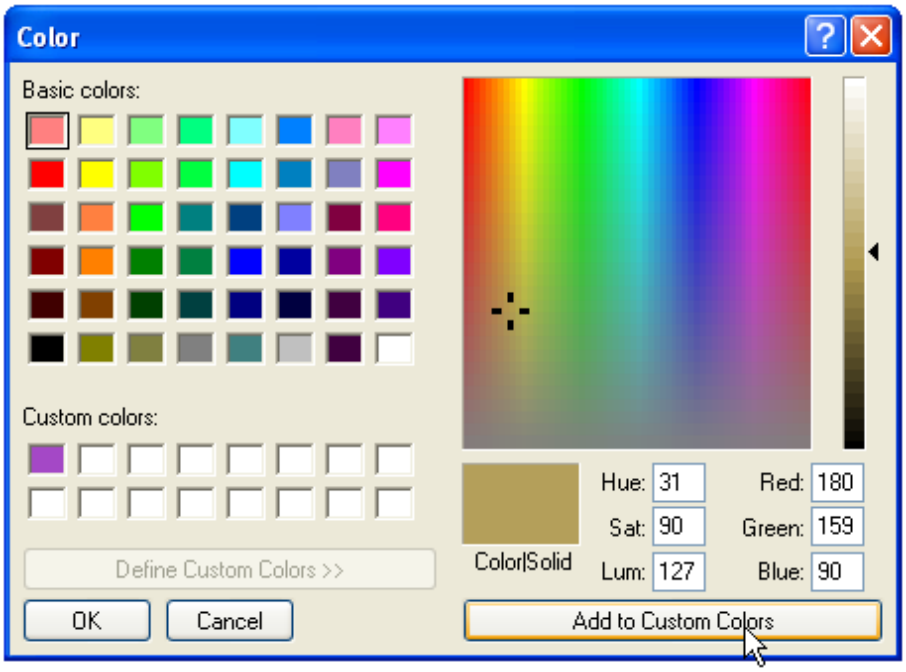


1. In the View > Options menu, click the 'Colours...' option to open the Colours display.
2. Use the Spin button to increase or decrease the intensity of the watermark. As the number in the field increases the watermark intensifies, while as the number decreases the watermark progresses towards matching the selected worksheet colour.
3. Click on OK to confirm the change and to close the window.

CUSTOM COLOURS

Clicking on the 'Custom...' key causes the basic colour swatch to appear, allowing the user to select one of 48 standard colours for the selected item.

Clicking on the 'Define custom colours>>' key allows the user to define new colours which can then be selected for use.



A new colour can be defined by clicking in the 'rainbow', by setting Hue, Saturation and Luminosity values or by entering Red, Green and Blue values. The 'intensity' of the selected colour can be varied using the slider control (arrow head) near the right-hand edge of the display.

Once the required colour (as shown in the Color/Solid block) has been defined, clicking on 'Add to Custom Colors' saves the colour to the 'Custom colors' swatch.

Click on 'OK' to close the Customise colours display.

20.15.5 Customise LINtools Toolbars

LINtools permits the user to customise the Toolbars, allowing the user to create, edit and inspect the Commands and Toolbars or add functionality to existing Toolbars.

To access the dialog:

- Select **View** > **Options** > **Customize Toolbars**.

Toolbars page

The **Toolbars** page of this dialog specifies the appearance of the Toolbars.

Commands page

The **Commands** page of this dialog specifies the appearance of the Toolbars.

CUSTOMISE TOOLBARS DIALOG - TOOLBARS PAGE

The **Toolbars** page of this dialog specifies which Toolbars are displayed and permits the user to create new Toolbars. To access the dialog:

- Select **View** > **Options** > **Customize Toolbars** in the editor window.

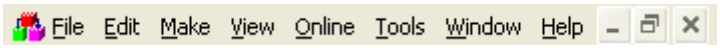
Click the **Toolbars** tab.

TOOLBARS

Lists all LINtools, both default and customer created, Toolbars.

- Menu bar

Set the checkbox ☑ to include this toolbar in this, and all further LINtools windows.



- File

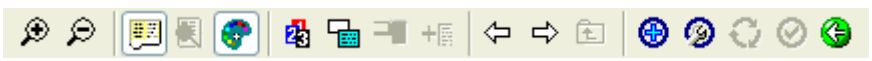
Set the checkbox ☑ to include this toolbar in this, and all further LINtools windows.

- Make

Set the checkbox ☑ to include this toolbar in this, and all further LINtools windows.

- View

Set the checkbox ☑ to include this toolbar in this, and all further LINtools windows.



- Misc

Set the checkbox ☑ to include this toolbar in this, and all further LINtools windows.

- Tools

Set the checkbox ☑ to include this toolbar in this, and all further LINtools windows.

APPEARANCE

Show ToolTips

Set the checkbox  to hide ToolTips when the cursor hovers over a button.

Cool Look

Set the checkbox  to show each button as an individual entity.

Large Buttons

Set the checkbox  to show each button as twice the size with the text caption.

New

Allows the user to create a custom toolbar.

Reset

Clicking on Reset causes the relevant 'built-in' toolbar to be returned to its default content. The text 'Reset' changes to 'Delete' if a custom toolbar is selected.

Delete

Allows custom toolbars to be deleted. The text 'Delete' changes to 'Reset' if a built-in toolbar is selected.

TO CREATE A CUSTOM TOOLBAR:

1. Click 'New...' in the 'View' > 'Options' > 'Customize Toolbars' display to show the 'New Toolbar' name dialogue.
2. In the Toolbar name box, type a name for the new toolbar and click on 'OK'. The new (empty) Toolbar appears, with the typed-in name.
3. To add a Command icon or Menu to the new Toolbar, click the Commands tab. In the Categories box, click a category for the first required command. Select the Command or Menu from the Commands box and drag it to the displayed toolbar. Repeat as necessary.
4. When all the required buttons and menus have been dragged to the new menu, click Close.

RESET/DELETE

To delete a custom Toolbar,

1. In the 'View' > 'Options' > 'Customize Toolbars' display, select the custom toolbar to be deleted.
2. Click 'Delete' to delete the toolbar (without confirmation).

Note: The text 'Delete' appears on the button only when a customised toolbar is selected. Built-in Toolbars cannot be deleted, and for such toolbars, the text reads 'Reset'. The Reset button restores the Toolbar to its default buttons, menus, and submenus.

OK

Click to close the Customise Toolbar dialog saving any changes.

Cancel

Click to close the Customise Toolbar dialog without changing anything.

Help

Click to see help topic.

CUSTOMISE TOOLBARS DIALOG - COMMANDS PAGE

The **Commands** tab allows the user to specify which Commands are displayed and permits the user to arrange the Command Icons as required. To access the tab:

- Select View > Options > Customize Toolbars in the editor window.

Click the Commands tab.

Categories

Lists all LINtools default Command categories:

File, View, Make, Misc, Tools, Menu

Clicking on a category displays all tool buttons or menus associated with the selected category.

Buttons

Shows all buttons associated with the selected category.

Select a button the reveal a brief description of its function, or drag to any Toolbar for use in LINtools.

Description

Shows a brief description of the selected button.

20.15.6 Customise LINtools Settings

Use the Settings dialogue to specify what the Editors and the Template Palette will display, and the required effect on Tags and TagNames:

1. Select **View** > **Options** > **Settings…**.
    - Alternatively, click in the **Block Template Palette** to display the context menu and select **Settings…**.
2. Click the appropriate tab.
    - The Editor page of this dialog specifies some wiring and display features of the function block **Editor** window.
    - The Palette page of this dialog specifies the style and features of the Block Template Palette.
    - The Tags page of this dialog specifies a required effect on Tags with unresolved aliases and changed block TagNames.
    - The Advanced page of this dialog is used to enable Intellectual Property Protection.

SETTINGS DIALOGUE - EDITOR PAGE

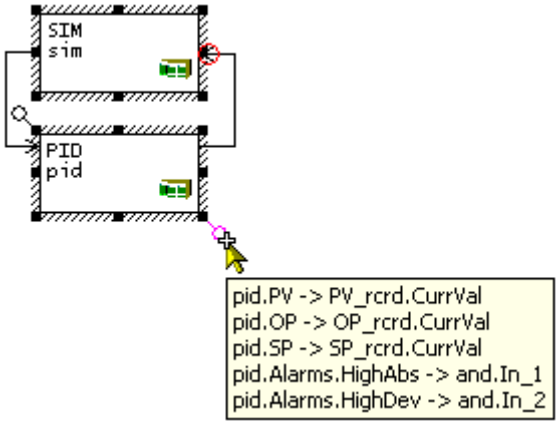
The **Editor** page of this dialog specifies some wiring and display features of the function block **Editor** window. To access the dialog:

- Select **View** > **Options** > **Settings...** in the **Editor** window.
- Click in the **Block Template Palette** to display the context menu and select **Settings...**.

Click the **Editor** tab.

Show Compound Block External Connections

Set the checkbox  to display external connections when viewing the contents of a compound. All external connections appear as 'lollipop' symbols attached to the connected function blocks - at the top left corner for inputs and the lower right corner for outputs (see image below). Hovering the cursor over the end of a 'lollipop' opens a window listing the connection sources and destinations.



Show Tooltips on Wires offline

Set the checkbox ☑ to display a yellow 'tooltips' window listing all connection sources and destinations in a wire, when the cursor is hovered over the wire.

Note: Only works for offline displays. In the Database Monitor, the 'tooltip' shows the name(s) and current value(s) of the field(s) being transmitted by the wire.

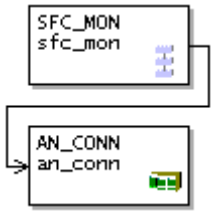
Show Comment Text Tooltips

Check the box to display a yellow 'tooltips' window containing the Comment Text associated with the selected block or step. This is indicated by the Comment tab Text symbol in the upper right corner of the function block or step.

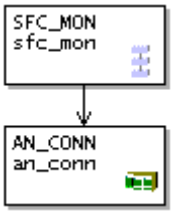
Wire: Right out - Left In

This option specifies how you want wiring to appear as it enters and leaves a function block.

- Set the checkbox ☑ to force wires to leave a function block from its right-hand edge, and enter a function block via its left-hand edge - see image below:



- Leave the box unchecked to allow wires to enter and leave blocks at the most convenient edges - see image below:



New File Options

Use Large Worksheets

This option specifies the Worksheet size you want.

- Set the checkbox ☑ to specify the use of an extended worksheet.

OK

Click to apply your Editor selections.

Cancel

Click to close the **Settings** dialog without changing anything.

Help

Click to see this help topic.

SETTINGS DIALOGUE - TAGS PAGE

The **Tags** page of this dialog specifies a required effect on Tags with unresolved aliases and changed block TagNames.

To access the dialog

- Select **View** > **Options** > **Settings...** in the Editor window.
- Click in the **Block Template Palette** to display the context menu and select **Settings...**.

Click the Tags tab.

Unresolved Aliases

- Leave Tags

Click the radio button to specify that no action will be taken with any unresolved Tag in the Tag Table.

- Detach Tags

Click the radio button to specify that any unresolved Tag in the Tag Table will be unattached from the current point, and can be re-attached to another point in the system.

- Delete Tags

Click the radio button to specify that any unresolved Tag in the Tag Table will be deleted and removed.

When block Tag Name is changed

- Match LIN Name if possible

Click the radio button to permit a TagName of up to 8 characters, that will allow the unique network LIN Name to match the TagName.

- Do not change LIN Name

Click the radio button to permit a TagName of up to 16 characters that will retain the unique network LIN Name.

OK

Click to apply your Template Palette selections.

Cancel

Click to close the Settings dialog without changing anything.

Help

Click to see this help topic.

SETTINGS DIALOG - PALETTE PAGE

The **Palette** page of this dialog specifies the style and features of the Block Template Palette. To access the dialog:

- Select **View** > **Options** > **Settings...** in the editor window.
- Click in the **Block Template Palette** to display the context menu and select **Settings...**.

Click the Palette tab.

Categorised Tab style

- Tree

Click the radio button to select the 'Tree' view of the block templates. Block categories and function blocks are displayed in a 'Windows Explorer' style hierarchy. Click the '⊞' and '⊟' boxes to reveal or hide individual function blocks.

- List

Selects the 'List' view of the block templates. Categories and function blocks are displayed in separate sub-windows, which can be configured to show as large or small icons. Right-click in a sub-window to reveal a context menu for selecting options.

Enable Tabs

- Alphabetical

Check the box to include an 'Alphabetical' tab in the Template Palette. Clicking this tab displays all the function blocks in the template in alphabetical order (for both 'Tree' and 'List' views).

- Recent

Check the box to include a 'Recent' tab in the Template Palette. Clicking this displays a list of function blocks placed this session, (for both 'Tree' and 'List' views).

OK

Click to apply your Template Palette selections.

Cancel

Click to close the Settings dialog without changing anything.

Help

Click to see this help topic.

SETTINGS DISPLAY - ADVANCED PAGE

The **Advanced** page provides the ability to prevent individual database files, sequence files and action files from being duplicated. To access the display:

- Select **View** > **Options** > **Settings...** in the **Editor** window.
- Click in the **Block Template Palette** to display the context menu and select **Settings...**.

Click the **Advanced** tab.

Intellectual Property Protection

Set the checkbox  to enable the **Save as password encrypted file** checkbox on the 'Save As' dialog.

20.15.7 Customise LINtools Worksheet

In a configuration the strategy can become quite complex and may require many different Database function blocks and/or Sequence Steps. If you find that the default size worksheet is becoming too cluttered and it is difficult to read the processing required by the LIN Database, or a route through a Sequence, this command will increase the size of the worksheet. For FBD and SFC this means the height and width are doubled, and for Ladder Diagrams [LD] the number of rungs are doubled from 24 to 48, but the width remains at 8 elements per rung.

Note: New large worksheet graphics files will not be compatible with the previous versions of LINtools.

TO CONVERT TO A LARGE WORKSHEET:

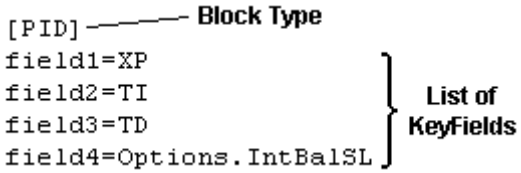
Simply select the **Convert to a Large Worksheets** command from the **View** > **Options** menu to increase the size of the currently opened worksheet.

Note: If all future Worksheets to be created on a large worksheet configure the Use Large Worksheets option in the Settings dialog.

See Also: Settings dialog, Use Large Worksheets

EDITING KEYFIELDS.INI FILE

KeyFields are defined in the 'KeyFields.ini' file in the **Program Files** >...> **LINtools** > **libs** directory. This contain entries for each Function Block Type in the following form.



Where sections are of Function Block Type ([Block Type]), and KeyFields (fieldn=..n), list the fields to be displayed.

To edit the KeyFields.ini file:

1. Open the KeyFields.ini file using Notepad or any other text editor.
2. Find the Function Block Type to be edited.

   Add or delete the references as required.
   - Add KeyFields using the fieldn=<field name> format.
   - Delete KeyFields by removing the line of text and renumbering the `field<n>=Options`.

3. Save and close to complete these instructions.

20.16 PROTECT INTELLECTUAL PROPERTY

LINtools permits you to password protect database files, sequence files and action files. Recipe files can be password protected using the Recipe editor.

20.16.1 To password protect files:

1. Select **View** > **Options** > **Settings…**.
2. Click the Advanced page. This is used to manage the Intellectual Property Protection configuration.
   - Set the **Offer option to encrypt files on Save As** checkbox ☑ and close the dialog. This enables the **Save as password encrypted file** checkbox on the Save As dialog allowing password protection to be configured for this and any further file.

Note: Intellectual Property Protection will remain enabled until the Offer option to encrypt files on Save As is disabled (☐).

3. Select **File > Save as** command to display the Save as dialog. The **Save as password encrypted file** checkbox is now available.

Note: LINtools cannot open a protected file without the correct password. Protection and storage of passwords is the user's responsibility. The manufacturer cannot recover lost passwords or open protected files.

i. Set the **Save as password encrypted file** checkbox  and press Save. This launches the **Set encryption key** dialog. This dialog is used to configure the identifier and password for this file.

ii. Enter the key identifier and Password. A key identifier can be a keyword (or memorable word) that is used to help associate the encrypted file with the corresponding password. It is also used by the instrument to identify which password is required to access the protected file.

A new key identifier can be configured or an existing key identifier can be used. When the **Use a new encryption key** is selected, enter the required key identifier. The configured password must then be confirmed to ensure that it the password used is as intended.

If **Use an existing encryption key** is selected, all existing key identifiers that have already been used this session are available from the drop-down list. This will automatically assign the corresponding password.

Note: The same key can be used by multiple files. Files in the strategy can be encrypted individually with different keys but a single instrument is limited to storing seven keys.

iii. Click on 'OK' to accept the key identifier and corresponding Password configuration.

Note: Select Online > *Manage Instrument Passwords* command to connect to the instrument and display a dialog listing all the keys currently stored in the instrument. Use this dialog to manage the keys currently stored in the instrument.

20.16.2 Manage the instrument passwords

A maximum of seven keys is permitted per instrument. When connected to the instrument, the **Instrument Passwords** dialogue provides a method of managing the available keys.

20.16.2.1.1 To manage the passwords of a 'Live' Instrument

1. With a Database file, .dbf, open select **Online > Manage Instrument Passwords** command. The **Instrument Passwords** display appears, with a message that LINtools is attempting to connect to the instrument.

Note: If a database is not open in LINtools, a dialogue is displayed which can be used to specify the Port and Node of the instrument required.

2. Once connected general instrument information is displayed, including Instrument type and Version, Port, and Node address and a list of the keys currently stored in the instrument. Only key identifiers are shown, not Passwords.

3. Use the buttons below the listed key identifiers to manage the keys' configurations.

- Press the **'Add'** button to add a key to the instrument. A dialogue is displayed allowing a new key to be created or a key that has already been used during this session to be selected.

Note: If more than seven keys have been downloaded to the instrument, at least one key must be deleted before a new key can be added.

- Select a key Identifier that is not required and press the **'Remove'** button to delete that key from the instrument. Confirmation is required before the key is deleted.

Caution

A key cannot be retrieved after it has been deleted. Deleting a key prevents the instrument from opening files encrypted with that key. This may stop the instrument from operating correctly.

21 GETTING SPECIFIC HELP

21.1 LINTOOLS WINDOWS HELP

To get help in using LINtools Editor and Monitor windows, click the shortcuts:

Database Editor

Online Configuration

Specific Sequence Editor

Generic Sequence Editor

Sequence Action Monitor

Action Editor

I/O Configurator

Profibus Master Configurator

Data Recording Configurator

TO GET HELP ON INDIVIDUAL FUNCTION BLOCK FIELDS:

In the relevant **Block** tab, click on a field name or value and press<**F1**> to display a help window for that field.

A **Block** tab can be found:

- by highlighting the function block and opening the Object properties pane, or
- by using the Find function block facility.

GETTING 'WHAT'S THIS' HELP:

Sometimes you can see more specific help by clicking the Help toolbutton, then clicking on the object of interest.

Note: Be sure to click inside a window, not on its banner or docking bar.

21.2 LIN FUNCTION BLOCK HELP

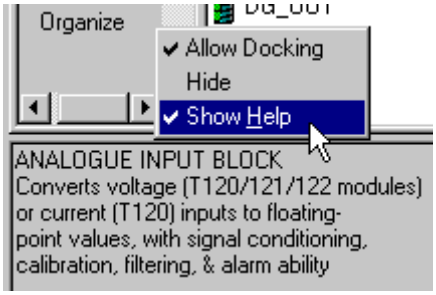
Help with a particular LIN function block can be obtained in two ways:

- a **concise** description of what the function block does, or
- a **comprehensive** description of the function block and all its parameters, via a link to the relevant section of the LIN Blocks Reference Manual in Online Books.

21.3 GETTING CONCISE FUNCTION BLOCK HELP

When you highlight a Function block in the Function Block Template Palette, a summary of what the function block does normally appears in the help pane below the palette.

- To hide or see this help information, right-click the template palette's banner, docking bar, or borders and select **Show Help** from the menu - see image below.



21.4 VIEWING THE LIN BLOCKS REFERENCE MANUAL

To access the Online Manual at the section dealing with a particular function block:

1. In the Database Editor (or Monitor) window, click on a field name or value in the relevant **Object properties pane** and press <**F1**> to display a help window for that field.

Editor and Monitor windows

2. At the bottom left corner of the help window is a **Books Online** button. Click the button to open a window showing the LIN Blocks Reference Manual (HA082375U003).

22 INDEX

C

T

Scan for local contents

Eurotherm Ltd
Faraday Close
Durrington
Worthing
West Sussex
BN13 3PL
Phone: +44 (0) 1903 268500
www.eurotherm.co.uk

HA263001U055 Issue 19 (CN35914)