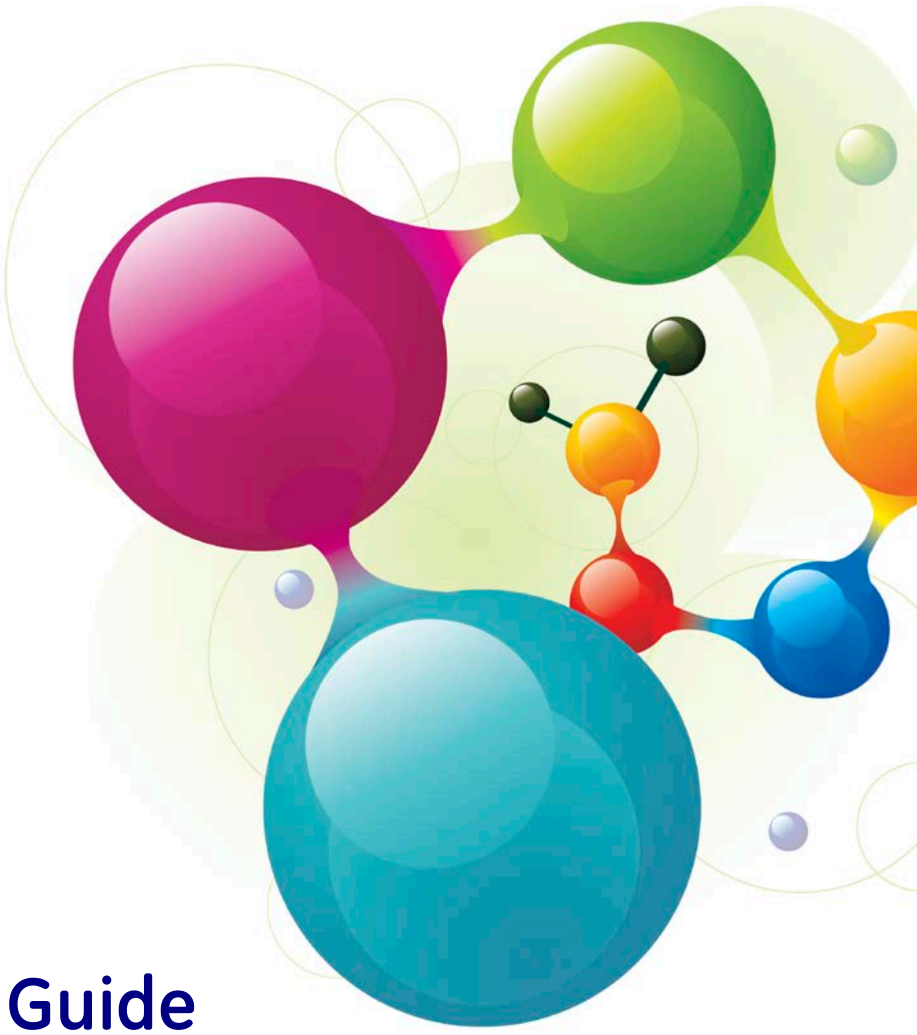
GE Energy



Configuration Guide

PowerOn*Fusion*

Data Utilities

Plant File Loader (PFL) Utility

G021-03-05-04-v513

Release v5

imagination at work

Copyright

©2011 General Electric Company. All Rights Reserved. GE and the GE monogram are trademarks and service marks of General Electric Company.
The Software Product described in this documentation may only be used in accordance with the applicable License Agreement. The Software Product and Associated Material are deemed to be "commercial computer software" and "commercial computer software documentation," respectively, pursuant to DFAR Section 227.7202 and FAR Section 12.212, as applicable, and are delivered with Restricted Rights. Such restricted rights are those identified in the License Agreement, and as set forth in the "Restricted Rights Notice" contained in paragraph (g) (3) (Alternate III) of FAR 52.227-14, Rights in Data-General, including Alternate III (June 1987).
If applicable, any use, modification, reproduction release, performance, display or disclosure of the Software Product and Associated Material by the U.S. Government shall be governed solely by the terms of the License Agreement and shall be prohibited except to the extent expressly permitted by the terms of the License Agreement.
The information contained in this document is subject to change without notice. The software described in this document is supplied under license and may be used or copied only in accordance with the terms of such license.
The information contained in this document is the exclusive property of General Electric Company except as otherwise indicated, and shall not be disclosed or reproduced in whole or in part without prior written consent from General Electric Company.
The information contained in this online publication is the exclusive property of General Electric Company, except as otherwise indicated. You may view, copy and print documents and graphics incorporated in this online publication (the "Documents") subject to the following: (1) the Documents may be used solely for personal, informational, non-commercial purposes; (2) the Documents may not be modified or altered in any way; and (3) General Electric Company withholds permission for making the Documents or any portion thereof accessible via the internet. Except as expressly provided herein, you may not use, copy, print, display, reproduce, publish, license, post, transmit or distribute the Documents in whole or in part without the prior written permission of General Electric Company.
The information contained in this online publication is subject to change without notice. The software described in this online publication is supplied under license and may be used or copied only in accordance with the terms of such license.

Trademarks

* Trademark of General Electric Company.

Oracle is a registered trademark of ORACLE Corporation
UNIX and Motif are registered trademarks of The Open Group

Other company or product names mentioned in this document may be trademarks or registered trademarks of their respective companies.

Change History

Date	Author	Version	Section	Change Description
27 May 2009	**MN**	**1.0a**	**All**	**CN 82578: Update document for PowerOn v5 release**
28 May 2009	MW	1.0	All	Approved.
10 November 2009	**MW**	**1.1a**	**All**	**CN 88885: Update document for PowerOn* v5.0.2 release.**
10 November 2009	MW	1.1a	11.4	TI 49293: Updated description of the Update Scan Point command.
19 February 2010	MAG	1.1	All	Approved.
14 May 2010	**DMC**	**2.0a**	**All**	**CN 105342: Update document for PowerOn* v5.1 release.**
14 May 2010	DMC	2.0a	6	WP 102009: Added cross-reference for single phase functionality.
11 June 2010	DMC	2.0	All	Approved.
12 April 2011	**MW**	**513a**	**All**	**CN 127789: Update document for PowerOn* v5.1.3 release.**
12 April 2011	DMC	513a	2, 2.2.1 & 8	CN 130990: Removed reference to RTAP.
19 August 2011	MW	513	All	Approved.

Contents

1. INTRODUCTION TO PLANT FILE LOADER (PFL) UTILITY 6
1.1 PFL Environment Requirements 6

2. PFL COMMAND LINE PARAMETERS 7
2.1 Examples 10
2.2 Error Reporting 10
2.2.1 Runtime Errors 10
2.2.2 Return Value 11

3. PFL FILE FORMAT 12

4. SELECTING COMPONENTS 13

5. SELECTING/BUILDING COMPONENTS 16

6. POPULATING COMPONENTS 18

7. SELECTING ATTRIBUTES 20

8. POPULATING ATTRIBUTES 21

9. APPLYING CTES 25

10. BUILDING ATTRIBUTES 26

11. RT INSTRUCTIONS 28
11.1.1 RTU Commands 28
11.1.2 FEP, Line and RTU Relationship Commands 28
11.1.3 Card Commands 29
11.1.4 Scan Commands 30
11.1.5 Control Commands 33
11.1.6 Properties 35
11.1.7 Linking Scan Points to Components 36
11.1.8 Associating Scan Points With Control Points 36
11.1.9 Adding Control Definitions 37
11.1.10 Default Card Command 37

12. ICCP INSTRUCTIONS 38
12.1.1 DL Association Commands...... 38
12.1.2 DL Control Center Commands...... 39
12.1.3 DL Dataset Commands 39
12.1.4 DL Device Commands...... 40
12.1.5 DL Domain Commands...... 40
12.1.6 DL Point Commands 41
12.1.7 DL Transferset Commands 42
12.1.8 Doc Interpretation Incoming Commands 43
12.1.9 Doc Interpretation Outgoing Commands 43
12.1.10 ICCP Control Table Commands...... 43
12.1.11 ICCP Interpretation Incoming Commands...... 44
12.1.12 ICCP Interpretation Outgoing Commands...... 44
12.1.13 ARP Dlpoint Commands 44
12.1.14 ARP Profile Commands...... 45

13. OTHER INSTRUCTIONS 46

14. EXAMPLE PFL INPUT FILES 47
14.1 Example 1...... 47
14.2 Example 2...... 48
14.3 Example 3...... 50

15. PFL COMMANDS IN NUMERICAL ORDER 51

Tables

Table 1: PFL Commands in Numerical Order 54

1. Introduction to Plant File Loader (PFL) Utility

The plant_file_loader (PFL) utility is used to select/build PowerOn* components and select/build and populate PowerOn attributes. The utility does this by parsing a datafile containing a list of instructions in 'tag'-'value' form.

The datafile directs the utility to the requested attributes, which can then in turn be populated with the specified data.

The utility is a command line-driven UNIX® or NT executable that can be run from an NMS server or a UNIX/NT client.

The utility populates or creates attributes by generating PowerOn transactions that are distributed to all NMS and SCADA servers.

1.1 PFL Environment Requirements

The utility resides in the default installation directory for the PowerOn product, which is normally /users/bin. This directory should be included in the $PATH environment variable.

The following environment variables should also be defined:

- ENMAC_USER=enmac
- ORACLE_SID=nms

2. PFL Command Line Parameters

The usage for the utility is as follows:

```
load_plant_file [-a alias] [-b] [-B] [-L path/alias]
                [-N component name] [-T template alias]
                [-c] [-C] [-p prefix] [-U]
                [-h] [-H]
                [-o o/p info lvl] [-O option]
                [-n attribute name] [-t] [-w seconds]
                [-u RDBMS user] [-s RDBMS server]
                [-D RT digital plant type]
                [-A RT analogue plant type]
                [-R RT telecontrol plant type]
                [-S RT scan value data type]
                [-i RT associated value type]
                [-d RT deadband type]
                [-v RT deadband value]
                <file> [<file>...]
```

The parameters are described below:

- -a alias

  Only load attributes for the component with the specified alias (that is, ignore the attributes for all other components).

- -b

  A component is searched for. If it is not found the utility attempts to create/build a single component.

  This creation could take a number of forms. It could be through the cloning of a template component, or if a template component is not specified, or if the specified template is not found then the PFL will build a basic component from scratch. Note that the -b signifies a 'single creation/copy' of a component. Other parameters can be set to tailor the building of components, but -b is the basic build parameter that must be set before the utility will build any components. If -b is not set on the command line and the search for a specific component fails, the utility will not attempt to create a component, but will move on.

- -B (Format: -b -B)

  The -B flag is an additional option that can be used in conjunction with the -b flag. It implies the copying of a 'tree'—not just a single point. When you set the –B parameter you should also specify a template component. If a template component is not specified or it is not found, a warning is generated and the -B flag is ignored. If a template is not specified then there is no possible tree structure to copy, so the -B parameter is inappropriate.

  Note: The -b flag is always required to build components, and implies the 'copy single' function. The addition of the -B flag will build the tree.

- -T alias (Format: -b –T alias)

  This parameter is required to set a template from which possible new components may be cloned. The template component is specified by its alias. It is invalid to set the -T parameter without setting the -b parameter on the same command line.

  The specified template is cloned each time a component in the datafile is found not to exist. There is an exception to this; the user may specify an 'override template' as part of a 'tag'-'value' in the datafile. Hence the component would be cloned from the override template and not the 'general' template specified on the command line. Refer to Select a Component and if not found, try to Build a Component for details.

- -L alias –N component name (Format: -b –L alias –N component name)

  The above two parameters are explained together, as it is invalid to set one and not the other. If the search for a component fails and the user wishes to create a component (through the setting of the -b parameter) then the utility needs to know where to create the new component and what to call it. The –L parameter sets the location of where the new component will be placed though specifying the parent of the new component. It is under this parent that the new component is located. The -N parameter sets the name of the new component.

  There are several ways to specify the location of the parent component. Refer to <path to parent> for details.

  Note: The location and component name specified is ignored if these build parameters are provided with the Build 'tag'-'value' in the PFL datafile. Refer to Select a Component and if not found, try to Build a Component for details.

- -c

  If the attribute is not present in the component, then the attribute is searched for in the component's template and if found, it is cloned. There are two possible templates that can be searched for. If an instruction ('tag') in the datafile requested the use of an override template, then search for the override template. Otherwise, if the component was cloned from a template then use this template.

  If the component was not cloned from a template and no override template exists the utility skips the population of the current attribute. See below if both -c and -C are specified.

- -C

  If the attribute is not present in the component, the attribute is built. The information detailing the new attribute is obtained from a series of previously processed 'tag'-'value' pairs in the datafile. When the -C flag is employed on its own, no search for a possible template will occur.

  If both -c and -C are on the command line then -c takes precedence. That is, the template attribute is searched for first and if that fails, only then will the attribute be created.

  Note: If a selected attribute is not found in the database and no build information is provided or the information provided is invalid, then for the case of a **vector**, **table** or **text** attribute build, an error or warning is raised and the utility moves on to the next attribute. If however the build type is **scalar** and either the build information is not present or invalid then the utility attempts to create a 'Not Applicable' scalar attribute. If this fails the utility raises an error or warning and moves on to the next attribute.

- -h

  Displays Help.

- -H

  Displays more detailed Help.

- -p prefix

  Prefix each alias populated with a prefix character. This option is most likely used with a PFL file generated by the **generate_plant_file** (GPF) utility, so that populating the alias will not be rejected if the original alias is still in the database.

- -P

  Update patched components as well as the real-world components.

- -n

  Only load attributes with the specified attribute name. That is, ignore attributes that do not match this name.

- -o output information level

  The amount of information to output (useful for debugging and logging):

  - -o1 show current component
  - -o2 same as o1 and show current attribute
  - -o3 same as o2 and show value and definition that is being loaded
  - -o4 show as much information as possible. This generates large amounts of text.

- -O option

  Special options.

  - -Oa customer specific
  - -Oc customer specific
  - -Oh displays Help on the special options

- -t

  Test mode. Parses the datafile to check for errors, but does not generate the transactions, so the attributes in the PowerOn database are not updated.

- -U

  Update attributes if value is different. Compares the following values in the PFL file with that in the database:

  - Scalar Attributes
  - Vector Attributes
  - Table Attributes
  - Component Header
  - Component Alias
  - Attribute Header

  If they are different then the update is performed, otherwise a message is displayed to the user. Note that this does not apply to RT functionality.

- -w wait period in seconds (default: 0.0 seconds)

  The time delay, in seconds, to the nearest microsecond, to wait between executing transactions. This parameter can be used to reduce system load on the NMS server and RT.

- -u RDBMS user

  By default, the utility connects to the host's preferred server and user. Normally it does not matter which server it is connected to because all updates are carried out as transactions. However, sometimes it may be required to connect to a particular user/server combination and these parameters provide this facility.

- -s RDBMS server

  See -u parameter above.

- -D RT digital plant type

  Default RT digital plant type. The user comment of the plant type to use if one hasn't been provided when creating scan points.

- -A RT analogue plant type

  Default RT analogue plant type. The user comment of the plant type to use if one hasn't been provided when creating scan points.

- -R RT telecontrol type

  Default RT telecontrol type. The user comment of the telecontrol type to use if one hasn't been provided when creating control points.

- -S scan value data type

  Default RT scan value data type. The value to use for scan value data type when creating scan points.

- -I associated value data type

  Default RT associated value data type. The value to use for assoc val data type when creating scan points.

- -d deadband type

  Default RT deadband type. The value to use for deadband type when creating scan points.

- -v deadband value

  Default RT deadband value. The value to use for deadband value when creating scan points.

2.1 Examples

To check the PFL datafile 'file01' for errors and to direct all errors and information to the screen and file '/tmp/pfl.log'. For K-shell users:

```
load_plant_file -o3 -t file01 2>&1 | tee pfl.log
```

To load the above file for real, but this time only direct errors and information to a file:

```
load_plant_file -o3 file01 > /tmp/pfl.log 2>&1
```

2.2 Error Reporting

2.2.1 Runtime Errors

Errors are directed to stderr. NMS transaction errors are logged in the system error log on the NMS server.

2.2.2 Return Value

If the datafile was loaded successfully, then 0 is returned, otherwise non-zero is returned.

3. PFL File Format

The datafile contains a list of instructions to direct the utility to attributes and then to load the attributes with a specified value. The file is a text file and each instruction occupies two lines.

The first line has a numeric value which represents the command to execute. The second line contains any data required for the command. The last instruction must be:

```
0
ENDSEC
```

This denotes the end of the file.

The instruction types can be categorised as follows:

- Selecting Components
- Selecting/Building Components
- Populating Components
- Selecting Attributes
- Building Attributes
- Populating Attributes
- Applying CTEs
- ICCP Instructions
- RT Instructions

Once a component is selected, all component population instructions and attribute addressing refers to the selected component.

Once an attribute is specified, all attribute population instructions refer to the selected attribute.

4. Selecting Components

By default the selected components are always components in the real world, so if a component is patched, only the real-world component is selected. However, patched versions of the component will also be selected if the '-P' option is specified in the command-line.

Select a Component by Alias

Command: 1

Data: <the alias of the component to select>

This is the fastest method of selecting a component.

Select a Component by Relative Pathname

Command: 1001

Data: <path to component>[<, attribute name, attribute value>]

Relative addressing is achieved by specifying a path to the component using component names. Each component name is separated by a semi-colon. The path may be given relative to the ROOT point by giving ':' as the first character; relative to an alias by giving the alias as the first portion of the path specification, or relative to the current root point by specifying '.' as the first portion of the path specification.

For example, the following are possible:

```
:EHV:11kV:Baillieston:Feeder 1:CB
Brec:Feeder 1:CB
.:CB
```

Note: An alias may be specified by itself, in which case this command is the same as command 1.

If all three parameters are given, an attribute containing the specified value is searched for in a branch. If a single component in the branch contains the attribute with that value, then that component is selected. If more than one component in the branch contains the attribute with that value, an error is reported. The first parameter specifies the branch to search (this component is included in the search). The second parameter contains the name of the attribute to search for and the third parameter contains the value of the attribute to match with.

Note: If the branch to be searched is evaluated to be ROOT, then an error is generated and the search is not performed.

Select a Component by Component Class and Specific Attribute

Command: 1006

Data: <component class>,<attribute name>,<attribute value>

This addressing method is used for selecting a particular component in a set of components that share the same PowerOn 'component class' but are uniquely identified by an attribute. The first parameter gives the PowerOn component class index (numeric), the second parameter gives the attribute to search for and the third parameter specifies the value to match with. If more than one component matches the specification, an error is generated.

For example:

```
1006
```

```
31,ICOND Network Number,567
```

will select the component that has a component class of 31 and has an attribute called 'ICOND Network Number' with a value of '567'.

Select a Component and if not found, try an Alternative Component

Command: 1007

Data: <first component specifier> | <alternative component specifier>

Both 'first component specifier' and 'alternative component specifier' have the same syntax as command 1001, that is:

<path to component>[<, attribute name, attribute value>]

A component specified by 'first component specifier' is searched for first and if found, this component is selected. If the first component specifier is not found, then the alternative component specifier is searched for. An error is only generated if a component cannot be found for either specifier.

Select a Component by Alias Wildcard

Command: 1100

Data: <the alias of the component to select>

All components that fulfil the wildcard specification are selected. The Oracle® wildcards are as follows:

- % represents zero or more characters
- _ represents any one character

For example:

```
1100
Asset 2000%
```

will select all components that have an alias beginning with 'Asset 2000', for example 'Asset 2000 - A', 'Asset 2000 CB' and 'Asset 2000'.

Specifying a Current Root Point

Command: 1009

Data: <path to component> [<, attribute name, attribute value>]

To allow relative access from a point other than ROOT, the current root point can be set using command 1009.

The syntax for 'path to component' is the same as that described for command 1100, except if 'path to component' is '$', then the current root point is set to the currently selected component.

If a single parameter is given, the current root point is set to this parameter.

If all three parameters are given, an attribute containing the specified value is searched for in a branch. If a single component in the branch contains the attribute with that value, then the current root point is set to that component. If more than one component in the branch contains the attribute with that value, an error is reported.

The first parameter contains the alias of the branch to search (this component is included in the search), the second parameter contains the name of the attribute to search for, and the third parameter contains the value of the attribute to match with.

Note: If the branch to be searched is evaluated to be ROOT, then an error is generated and the search is not performed.

5. Selecting/Building Components

Select a Component and if not found, try to Build a Component

Valid formats are as follows:

Command: 1101

Data: <alias of component to select>,<path to parent>,<component name>,[<override template>]

Command: 1101

Data: <alias of component to select>,<path to parent>,<component name>

Command: 1101

Data: <alias of component to select>

Command: 1101

Data: <alias to select> ,<path to parent> ,<component name> , [<override template>]

This command is dependent primarily on the command line parameter -b. If in addition the -B parameter is set, the command produces slightly different results, as described below:

<alias of component to select>

The utility searches for the component specified by this alias and selects the component if found. If this component is not found and if the -b command line parameter has been set, then the utility attempts to build a component. If in addition the -B parameter is set the utility attempts to build a tree structure instead of a single component.

<path to parent>

For the component creation you use the <path to parent> to specify where in the database to place this new component.

The <path to parent> is specified using relative addressing. Relative addressing is achieved by specifying a path to the component using component names. Each component name is separated by a semi-colon. The path may be given relative to the ROOT point by giving ':' as the first character; relative to an alias by giving the alias as the first portion of the path specification or relative to the current root point by specifying '.' as the first portion of the path specification.

For example, the following are possible:

```
:EHV:11kV:Baillieston:Feeder 1:CB
Brec:Feeder 1:CB
.:CB
```

You can specify the parent component directly by its alias, this being the simplest method.

<component name>

The <component name> is the name that would be given to a new component if one is to be created. It can also be referred to as the 'point name'.

<override template>

The [<override template>] value is optional, but the <path to parent> and the <component name> are both mandatory.

If the override template is provided and a search for a component alias fails then the utility attempts to clone from this template. The template is specified using its alias. However if an override template alias is not provided, or if the utility's search of the database for this template fails, then the utility checks the command line for a 'command line template'.

Note: This override template only takes effect for one component selection. On the next component selection using this command, if an override template alias is not given then the utility automatically uses (if present) the template alias on the command line.

General Notes

If command 1101 is encountered and the -b command line parameter is not set then the command acts as a command 1, ignoring the <path to parent> and the <component name>. This means that the functionality only attempts to select the component alias specified.

If this command is encountered and -b has been set but you fail to provide any information in the datafile following the alias of the component you want to select, then the utility looks to the command line. If you have set the -L and -N parameters with their appropriate arguments then this data is employed if a component build is required. However, note that the name and parent path provided on the command line are only applicable for one component build.

For example, the PFL datafile below is processed with the following command line.

```
Command line: load_plant_file –b -L ALIAS-281022-Q -N "component name"
Test.pfl
Test.pfl : 1101
     ALIAS-278834-Q
     1101
     ALIAS-298834-Q
     0
     ENDSEC
```

The command 1101 is encountered in the datafile but no additional parameters have been specified after the component alias.

The utility searches for the specified component alias, and if the search fails it considers the command line parameters and arguments. If a valid location and component name is obtained from the command line then the component is created. The utlity moves onto the next command. Again it encounters a 1101 command with only the desired component alias provided. This alias search also fails so the utility attempts to create the component using the data on the command line. This creation fails as under the parent location specified on the command line there already exists a component with the component name specified on the command line, from the previous build.

6. Populating Components

Store an Alias

Command: 4

Data: <component alias>

Stores the alias of the currently selected component. If the -p option has been specified the alias is prefixed with a character. For example, if the prefix character is 'x', an alias of '123DT40510' is populated as 'x123DT40510'.

Update a Component Header

Command: 7

Data: <property code, property value> [<property code, property value> ...]

This command allows the various properties of a component header to be altered. The valid property codes are as follows:

- 2: Component Name
- 3 : User Ref
- 4 : Current Priority
- 5 : Normal Priority
- 7 : Easting
- 8 : Northing
- 9 : Class
- 10 : Location
- 14 : Connectivity Class
- 15 : District Zone
- 17 : CE Operation
- 18 : Trace Class
- 19 : Substation Class
- 20 : Plant Ownership
- 23 : Naming

Note that additional options are available in single-phase networks. Refer to the *Single Phase Configuration Guide—PFL Single Phase Options* for details.

For example, to change the Component Name to 'test' and the Plant Ownership of a component to 'PS-Distribution', enter:

```
7
2, test, 20, 0
```

Note: In the example above, '0' represents 'PS-Distribution. This is because the Plant Ownership field (along with several others) in the component header is populated through a lookup. Entries in lookups are configured in the **Lookups** table. Each entry has a corresponding number specified in the LOOK_UP_REFERENCE field. It is this number that is entered in the PFL datafile.

7. Selecting Attributes

Select an Attribute

Command: 2

Data: <attribute name>

Selects the specified attribute in the currently selected component. If the attribute creation flags have been set on the command-line (-c and -C options), it is at this point when the attribute is created if it did not exist.

Select a Row of a Vector and a Table

Command: 61

Data: <row number>

Selects the vector entry or table row (for vector and table attributes respectively). Row numbers start from 1.

Select a Column of a Table

Command: 60

Data: <column number>

Selects the column number for a table attribute. Column numbers start from 1.

8. Populating Attributes

Store a Value

Command: 3

Data: <value>

Stores the value in the currently selected attribute. If the value contains '%s', this is replaced by a stored string. Refer to Store a String for Later Use in a Value of Definition for details. To store the value without interpretation, use Command 6, Store a Raw Value.

A Data Element (DE) type check is performed on the value to ensure that it is of the valid type. For example, if an attempt is made to place the value '12' into an attribute of DE type 'logical', an error is generated and the value is not stored.

Store a Raw Value

Command: 6

Data: <value>

Stores the value directly into the currently selected attribute without any interpretation.

A Data Element (DE) type check is performed on the value to ensure that it is of the valid type. For example, if an attempt is made to place the value '12' into an attribute of DE type 'logical', an error is generated and the value is not stored.

Store a Definition

Command: 1000

Data: <definition>

A definition is an RT Calculation Engine definition. This command stores the definition in the currently selected attribute. If the definition contains '%s', then this '%s' is replaced by a stored string. Refer to Store a String for Later Use in a Value of Definition for details.

Store a String for Later Use in a Value of Definition

Command: 1008

Data: <string to store>

This command stores the specified string for later use. The string may contain:

- %P which is replaced with the currently selected component point name, or
- %A which is replaced with the currently selected component alias

before the string is stored.

For example, if the currently selected component has an alias of 'ABC':

```
1008
Alias follows this: %A
```

will store 'Alias follows this: ABC' for later use.

To use the stored string for a value or definition, in the data part of the store instruction the inclusion of '%s' in the required place will insert the stored string at that point. For example:

```
3
Test *** %s ***
```

will store 'Test *** Alias follow this: ABC ***' in the currently selected attribute.

Select a Lookup

Command: 1002

Data: <lookup name>

The value or a definition of an attribute can be populated by selecting a value from a PowerOn lookup table. An item is picked out by specifying a lookup name and a lookup entry number. This instruction selects the lookup name, the item is only stored when the entry number is selected.

Store Lookup in Value

Command: 1003

Data: <lookup entry>

Fetches the lookup entry out from a previously selected lookup list and stores the item in the value of the currently selected attribute.

Store Lookup in Definition

Command: 1004

Data: <lookup entry>

Fetches the lookup entry out from a previously selected lookup list and stores the item in the definition of the currently selected attribute.

Store an Alarm Reference

Command: 62

Data: <alarm reference>

Stores the alarm reference of the currently selected attribute.

Update Attribute Header

Command: 5

Data: <property code, property value> [<,property code, property value> ...]

This command allows the various properties of an attribute header to be altered. The valid property codes are as follows:

- 1 : Attribute name
- 2 : Index
- 3 : Read group
- 4 : Write group
- 5 : Location
- 9 : Data Element (DE) type

- 10 : Alarm reference (this achieves the same as command 62)
- 12 : GRTV logging
- 13 : RDBMS archiving
- 14 : Event priority
- 15 : Attribute processing
- 16 : Protection level
- 17: CE Eval Mode
- 18 : Alarm Index
- 19 : Alarm Filter
- 20 : Source
- 21 : Identify
- 22 : Statistics profile
- 23 : RT calc periodicity

For example, to change the index of an attribute to '100' and the CE Eval Mode to 'Immediate':

```
5
2, 100, 17, 0
```

Note: In the example above, '0' represents 'Immediate'. This is because the CE Eval Mode field (along with several others) in the attribute header is populated through a lookup. Lookups are configured in the **Lookups** table. Each entry has a corresponding number, specified in the LOOK_UP_REFERENCE field. It is this number that is entered in the PFL datafile.

Update Attribute Table Size

Command: 1102

Data: <table size increase/decrease>, < fieldName0, fieldDeType0>,< fieldName1, fieldDeType1>,...

This command allows you to increase or decrease the Table Size of an attribute. The previously selected attribute must be of type TABLE for this command to execute successfully. The format of the command depends on whether you are increasing or decreasing the Table Size. An example of how to use the command to increase the Table Size is given below:

```
1102
3, New Column1, 2, New Column2, 2, New Column3, 2
```

Note that the DE Types for each field are specified using numerical values. These numbers are mapped to a specific DE Type through a lookup table. The above example would result in three new columns being appended to the previously selected table attribute.

The example below shows how to decrease the number of columns in the selected table.

```
1102
-3
```

Note that the <table size increase/decrease> value is a negative number and no new column names or column DE types are required. The above example would result in the last three columns of the selected table being removed. Note also that this command results in an update of the **Table Size** field in the attribute's header.

9. Applying CTEs

Component Template Extensions (CTEs) are used to extend the flexibility of component templates. Refer to the *Equipment Selector User Guide—Component Template Extensions (CTEs)* for details.

Note: The type of CTE functionality implemented is controlled by the OLD_CTE_TYPE system parameter.

Applying a CTE to a Component

Command: 1200

Data: <Process Label>

This command executes the actions specified by the CTE process, defined in the data arguments, on the selected component. The CTE process is defined in the **CTE Processing** table's PROCESS_LABEL field.

For example, if the CTE Processing table is configured with the following actions:

Process Label	Function Name	Function Data
Start_temp_wiz	APPLY_CTE	Actions1
Actions1	INSERT_ALL_ATTRIBUTES	
Actions1	UPDATE_COMPONENT_HEADER	.$CH{trace_class}
Actions1	CTE_QUESTION	Actions1_question

And the CTE Header table is configured as follows:

CTE Alias	Action
Cte_x	Actions1

A PFL file of the following form would be required:

```
1
test_component
1200
Actions1
0
ENDSEC
```

Note that the Process Label 'Actions1' rather than 'Start_temp_wiz' is used as this is the label that defines the actions to be performed.

In this example all the attributes from the CTE component 'Cte_x' are added by the INSERT_ALL_ATTRIBUTES action, and the Trace Class of the component is updated by the UPDATE_COMPONENT_HEADER action.

Note: The APPLY_CTE and CTE_QUESTIONS process functions are ignored in the PFL functionality. All sets of actions to be performed must be specified within the file.

10. Building Attributes

Build Attribute Type

Command: 66

Data: <attribute type>

This command sets the type for any future attribute creation. This attribute type is employed in all attribute creations until a new type is specified using the above command. Attribute types are specified in the **Lookups** table using the **Attribute Type** lookup. Refer to the *PowerOn Configuration Guide—Lookups* for details. Each attribute type has a corresponding integer value, defined in the LOOK_UP_REFERENCE field. It is this value that should be entered in the PFL datafile.

The example below would result in all future attribute creations being of type SCALAR, where '0' is the integer corresponding to the scalar attribute type.

```
66
0
```

Build Vector Size

Command: 64

Data: <vector size>

This command sets the vector size for any vector or table attributes that are created. When specifying the vector size in relation to a table attribute creation the size can be translated as the number of rows that will appear in the table attribute.

Build Table Size

Command: 65

Data: <table size>

This command sets the table size for any table attribute that is created. The term 'table size' refers to the number of columns that will appear in the table attribute.

Note: If you want to create a table attribute you must specify the table size parameter before you provide any field name or field DE Type data. The number of field names must match the table size that has been set with this command. This also applies to the number of field DE Types.

Build DE Type

Command: 63

Data: <DE Type>

This command sets the DE Type for any future attribute creation. DE Types are specified in the **Lookups** table. Each DE Type has a corresponding integer value, defined in the LOOK_UP_REFERENCE field. It is this value that should be entered in the PFL datafile.

Note: This command is only effective for a Scalar and Vector creation. A Text attribute creation does not require a DE Type. The DE Types for the individual columns in a table are specified using the Build Column DE Types command.

Build Column Names

Command: 8

Data: <ColumnName0, ColumnName1, ColumnName2, ...>

This command sets the column names employed in a future table creation.

Note: The number of column names specified must match the previously set table size. Refer to Build Table Size for details.

Build Column DE Types

Command: 9

Data: <ColumnDEType0, ColumnDEType1, ColumnDEType2, ...>

This command sets the column DE Types employed in a future table creation. Each DE Type has a corresponding integer value, defined in the LOOK_UP_REFERENCE field. It is this value that should be entered in the PFL datafile.

Note: The number of column DE Types specified must match the previously set table size. Refer to Build Table Size for details.

11. RT Instructions

This section describes the commands for populating the RT database tables. The following commands are covered:

- RTU Commands
- FEP, Line and RTU Relationship Commands
- Card Commands
- Scan Commands
- Control Commands
- Properties
- Linking Scan Points to Components
- Associating Scan Points With Control Points
- Adding Control Definitions
- Default Card Command

11.1.1 RTU Commands

Select an RTU

Command: 3000

Data: <rtu component alias>

Selects the current RTU. Any further scan/control point actions are made against this RTU.

Delete all Data

Command: 3001

Data: <rtu component alias >

This command deletes all data from the RT Scan system for an RTU.

11.1.2 FEP, Line and RTU Relationship Commands

Place a Line under a FEP

Command: 3002

Data: < fep alias >,< line alias >

This command places the Line under the specified FEP. If the line component already exists under another FEP it is moved to the specified FEP.

Place an RTU under a Line

Command: 3003

Data: < line alias >[,< rtu alias >][,<rtu parent alias>]

This command places the RTU under the specified Line. If the RTU component already exists under another LINE it is moved to the specified Line.

If <rtu alias> is not specified the command will place the currently selected RTU under the specified line.

If the optional parameter <rtu parent alias> is supplied then the <rtu alias> must also be specified. This will place the specified RTU under the specified parent RTU.

11.1.3 Card Commands

Create a Card

Command: 3100

Data: <type>, <protocol>,<description>

Pre-Requisite = Current RTU selected using command 3000, Select an RTU.

This command creates a new card in the currently selected RTU. The parameters have the following meanings:

Type

0 = input card

1 = output card

Protocol

This is the protocol of the card. This is an index into the RT Scan System protocol table.

Description

Free text description of the card.

Create a Tagged Card

Command: 3101

Data: <type>, <protocol>,<description>,<user tag>

Pre-Requisite = Current RTU selected using command 3000, Select an RTU.

This command creates a new card in the currently selected RTU and gives that card a user tag. The parameters are the same for command 3100, Create a Card, except for the last parameter which is a text description of the user tag.

Select a Card by User Tag

Command: 3102

Data: <user tag>

This command is used to select a previously created card by its user tag.

Select a Card by ID

Command: 3103

Data: <card ID>

This command is used to select a previously created card by its card ID.

Update a Card

Command: 3104

Data: <field>, <data>

Pre-Requisite = A Card is currently selected or has just been created.

This command is used to update information on a Card. The following field values are applicable:

- CONTROL_PROTOCOL

  Update the cards protocol index, data is the new protocol index.

- CARD_DESCRIPTION

  Update the card description

- CARD_USER_TAG

  Update the cards user tag.

Delete a Card

Command: 3105

Data: None required – leave a blank line

Pre-requisite = A card is currently selected or has just been created.

This command is used to delete a card.

11.1.4 Scan Commands

Create a Scan Point

Command: 3200

Data: <addr0>, <addr1>, <addr2>, <addr3>, <addr4>, <addr5>, <size>, <shift>, <desc> [,<plant type>]

Pre-Requisite = A Card and RTU are currently selected.

The parameters are as follows:

- **<addr0>** to **<addr5>** comprise the Card and Word address of the scan point
- **<size>** is the size in bits of the point
- **<shift>** is the bit position of the point in the Cards Word
- **<desc>** is a free text description of the scan point
- **<plant type>** is an optional value for the plant type. If this parameter is not supplied then the plant file loader will use the default command line parameter settings, digital plant type or analogue plant type dependant on the Card type.

Note: The following values are assumed by default when creating the point:

- **Control** is set to 1 (enabled)
- **Sval de type** is set to the command line parameter value
- **Assoc de type** is set to the command line parameter value
- **Rt deadband type** is set to the command line parameter value
- **Rt deadband** is set to the command line parameter value
- **Status** is set to 0.

Create a Tagged Scan Point

Command: 3201

Data: <addr0>, <addr1>, <addr2>, <addr3>, <addr4>, <addr5>, <size>,<shift>, <desc>, <user tag> [,<plant type>]

Pre-Requisite = A Card and RTU are currently selected.

This command creates a new scan point in the currently selected RTU and Card and gives the Scan Point a User Tag. The parameters are the same for command 3200, Create a Scan Point, except for the last parameter, which is a text description of the user tag.

Select a Scan Point by Tag

Command: 3202

Data: <user tag>

This command selects a scan point by its user tag.

Select a Scan Point by ID

Command: 3203

Data: <ID>

This command selects a scan point by its scan row ID.

Select a Scan Point by Address

Command: 3204

Data: <addr0> [,<addr1>] [,<addr2>] [,<addr3>] [,<addr4>] [,<addr5>] [,<size>] [,<shift>]

Pre-Requisites = RTU is selected.

Select a Scan point using its address. This is a variable length message allowing the operator to define as much data as needed to uniquely identify the point within the RTU. If more than one row is selected using this data then an error is given.

Update a Scan Point

Command: 3205

Data: <field>, <data>

Pre-Requisite = A scan point is currently selected or has just been created.

The values of the field parameter and the data in each case are as follows:

- SCAN_CONTROL

  Update the control field, 0 = disable and 1 = enable

- SCAN_VAL_DATA_TYPE

  Update the Scan Value Data Type

- SCAN_ASSOC_DATA_TYPE

  Update the Associated Value Data Type

- SCAN_DEADBAND_TYPE

  Update the Deadband Type

- SCAN_DEADBAND_VALUE

  Update the Deadband Value
- SCAN_ADDR0

  Update Address 0
- SCAN_ADDR1

  Update Address 1
- SCAN_ADDR2

  Update Address 2
- SCAN_ADDR3

  Update Address 3
- SCAN_ADDR4

  Update Address 4
- SCAN_ADDR5

  Update Address 5
- SCAN_SIZE

  Update the scan point size
- SCAN_SHIFT

  Update the scan point shift
- SCAN_PLANT_TYPE

  Update the scan points plant type
- SCAN_DESCRIPTION

  Update the scan points description
- SCAN_STATUS

  Update the scan point status, 0 = active, 1 = archived

Delete a Scan Row

Command: 3206

Data: None required – leave a blank line

Pre-requisite = A scan point is currently selected or has just been created.

This command is used to delete a scan row.

For example, to delete a scan row this is the file that should be run:

```
3000
<RTU_alias>
3203          /* to select the scan row */
<scan_row_id>
3206          /* to delete the scan row */
              /* blank line             */
0
```

```
ENDSEC
```

11.1.5 Control Commands

Create a Control Point

Command: 3300

Data: <addr0>, <addr1>, <addr2>, <addr3>, <addr4>, <addr5>, <desc> [,<control type>]

Pre-Requisite = A Card and RTU are currently selected.

The parameters are as follows:

- **<Addr0>** to **<Addr5>** comprise the Card and word address of the control point
- **<Desc>** is a free text description of the control point.
- **<control type>** is an optional parameter. If not set then the default Telecontrol control type, set via the command line, is used.

Note: The following values are assumed by default when creating the point:

- Control is set to 1 (enabled)
- Status is set to 0.

Create a Tagged Control Point

Command: 3301

Data: <addr0>, <addr1>,<addr2>,<addr3>,<addr4>,<addr5,<desc>,<user tag> [,<control type>]

Pre-Requisite = A Card and RTU are currently selected.

This command creates a new control point in the currently selected RTU and Card and gives the Control Point a User Tag. The parameters are the same for command 3300, Create a Control Point, except for the last parameter which is a text description of the user tag.

Select a Control Point by Tag

Command: 3302

Data: <user tag>

This command selects a control point by its user tag.

Select a Control Point by ID

Command: 3303

Data: <ID>

This command selects a control point by its control row ID.

Select a Control Point by Address

Command: 3304

Data: <addr0> [,<addr1>] [,<addr2>] [,<addr3>] [,<addr4>] [,<addr5>]

Pre-Requisites = RTU is selected.

Select a control point using its address. This is a variable length message enabling you to define as much data as needed to uniquely identify the point within the RTU. If more than one row is selected using this data then an error is given.

Update Control Point

Command: 3305

Data: <field>, <data>

Pre-Requisite = A Control Point is currently selected or has just been created.

The field parameter takes the following values:

- CONTROL_CONTROL

  Update the control field of the control point, values 0 = disable, 1 = enable
- CONTROL_ADDR0

  Update address 0
- CONTROL_ADDR1

  Update address 1
- CONTROL_ADDR2

  Update address 2
- CONTROL_ADDR3

  Update address 3
- CONTROL_ADDR4

  Update address 4
- CONTROL_ADDR5

  Update address 5
- CONTROL_PLANT_TYPE

  Update the control point plant type
- CONTROL_DESCRIPTION

  Update the control point description
- CONTROL_STATUS

  Update the control point status, values 0 = active, 1 = archived

Delete a Control Row

Command: 3306

Data: None required – leave a blank line

Pre-requisite = A control point is currently selected or has just been created.

This command is used to delete a control row.

11.1.6 Properties

Set a Property ID

Command: 3400

Data: <property ID>

This command sets the ID of the property for all following commands, this command is used regardless of whether we are adding a scan or control property.

Add a Value Property

Command: 3401

Data: <value>

Pre-Requisites = A property ID has been selected and either a control or scan point has been Selected.

This command adds the selected property (set by command 3400, Set a Property ID) to the last selected scan point OR control point. This command adds a Value property, where the value of the property is a number rather than a reference to another database point.

The following property fields are filled in by default:

- Type is set to 0, (0=value property, 1=reference property)
- Status is set to 0, (0=active, 1=archived)

Add a Reference Property

Command: 3402

Data: <component alias>,<attribute name>

Pre-Requisites = A property ID has been selected and either a control or scan point has been Selected.

This command adds the selected property (set by 3400, Set a Property ID) to the last selected scan point OR control point. This command adds a Reference property where the component alias and attribute name provide a point in the database from which to derive the value of the property.

The following property fields are filled in by default:

- Type is set to 1, (0=value property, 1=reference property)
- Status is set to 0, (0=active, 1=archived)

Update a Property

Command: 3403

Data: <field>, <data>[,<data2>][<data3>]

Pre-Requisites = A property ID has been selected and either a control or scan point has been Selected.

The <field> parameter takes the following parameters:

- PROPERTY_TYPE

  Change the property type (0=Property has a value, 1=Property is associated with an attribute value)

If the property type is changed to a value, then the <data2> parameter is the new value of the property. If the property type is changed to an attribute value, then the <data2> and <data3> parameters will be the component alias and attribute name for the attribute the property is associated with.

- PROPERTY_VALUE

  Update the value of a property

- PROPERTY_REFERENCE

  Update the reference of a property. In this case two data parameters are used, the Component Alias followed by the Attribute Name.

- PROPERTY_STATUS

  Update the status of the property, (0=The property is active for the point, 1=The property is archived for the point).

11.1.7 Linking Scan Points to Components

Add a Scan Reference

Command: 3500

Data: <component alias>, < scan attribute name>[, <associated attribute name>]

Pre-Requisites = A scan point has been created or selected.

This command adds a scan reference for a scan point. The parameters are as follows:

- **<component alias>** is the component alias of the object that is the scan point.
- **<scan attribute name>** is the attribute name of the attribute that will hold the scan value.
- **<associated attribute name>** is an optional parameter specifying the name of the attribute that will hold the associated value of the scan point.

The scan reference will automatically be created with a status of 0 (active)

Note: The system parameter PFL_LOG_SCAN_REF_INVAL_ATTR_ERR affects the functionality of this command, and determines if the scan reference is added if a component alias or attribute is not found. Refer to the *PowerOn Configuration Guide—System Parameters List* for details.

Delete a Scan Reference

Command: 3501

Data: <component alias>, < scan attribute name>[, <associated attribute name>]

Pre-requisite = A scan point is currently selected or has just been created. It is assumed the status of the component is 0.

This command is used to delete a scan reference.

11.1.8 Associating Scan Points With Control Points

Update a Scan Input Reference

Command: 3600

Data: <reference number>

Pre-Requisites = A scan point has been created or selected and a control point has been selected or created

This command associates an input point with a control point. The control point can be associated with a maximum of two input points. The parameter takes the following values:

- 1 = scan point reference 1
- 2 = scan point reference 2

11.1.9 Adding Control Definitions

Add a Control Definition

Control Definitions can be created in the same way that any attribute definition is handled in the plant file loader, however the following command allows the user to configure the control definition using user tags rather than control row IDs.

Command: 3700

Data: <user tag>, <control value>

Pre-Requisites = The component attribute has been selected in the usual way and the RTU has been previously selected.

This command inserts the following definition into the attribute:

```
CONTROL (<rtu_alias>, <control row>, <control value>)
```

The <control value> is taken directly from the command parameter; the <user tag> is interpreted to a Control Row automatically by the PFL utility.

11.1.10 Default Card Command

Set Default Card Type

Command: 3800

Data: <card type>

This command is used to set the default card type, which is used when creating Scan Points when no Plant Type has been provided with the command data. If the default card type is set to Digital, then the default digital type set on the command line is used. Otherwise, if the default card type is set to Analogue, then the default analogue plant type, also set on the command line, is used. The default card type is set to digital by default.

- 0 = default card type is a Digital card
- 1 = default card type is an analogue card.

12. ICCP Instructions

This section outlines the commands for populating the DL & ICCP database tables. The following commands are covered:

- DL Association Commands
- DL Control Center Commands
- DL Dataset Commands
- DL Device Commands
- DL Domain Commands
- DL Point Commands
- DL Transferset Commands
- Doc Interpretation Incoming Commands
- Doc Interpretation Outgoing Commands
- ICCP Control Table Commands
- ICCP Interpretation Incoming Commands
- ICCP Interpretation Outgoing Commands
- ARP Dlpoint Commands
- ARP Profile Commands

12.1.1 DL Association Commands

Create Association

Command: 5000

Data:<association name>, <server control center>, <client control center>, <association type>, <enabled>, <iccp priority>, <iccp routing>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp security>, <dual mode>, <ar name>

Pre-requisites: None

Creates a new entry in the ICCP.DLASSOCIATION table.

Update Association

Command: 5001

Data:<association name>, <server control center>, <client control center>, <association type>, <enabled>, <iccp priority>, <iccp routing>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp block1 support>, <iccp security>, <dual mode>, <ar name>

Pre-requisites: None

Updates the entry with the specified association name in the ICCP.DLASSOCIATION table.

Delete Association

Command: 5002

Data: <association name>

Pre-requisites: None

Deletes the entry with the specified association name in the ICCP.DLASSOCIATION table.

12.1.2 DL Control Center Commands

Create Control Center

Command: 5003

Data: <fep alias>, <control center name>, <control center abbr>, <local control center>, <enabled>, <dl access>, <ar name>

Pre-requisites: None

Creates a new entry in the ICCP.DLCONTROLCENTER table.

Update Control Center

Command: 5004

Data: <fep alias>, <control center name>, <control center abbr>, <local control center>, <enabled>, <dl access>, <ar name>

Pre-requisites: None

Updates the entry with the specified control center name in the ICCP.DLCONTROLCENTER table.

Delete Control Center

Command: 5005

Data: <control center name>

Pre-requisites: None

Deletes the entry with the specified association name in the ICCP.DLCONTROLCENTER table.

12.1.3 DL Dataset Commands

Create Dataset

Command: 5006

Data: <dataset name>, <domain reference name>, <transferset name flag>, <ds conditions flag>, <event code detected flag>, <transferset time flag>

Pre-requisites: None

Creates a new entry in the ICCP.DLDATASET table.

Update Dataset

Command: 5007

Data:<dataset name>, <domain reference name>, <transferset name flag>, <ds conditions flag>, <event code detected flag>, <transferset time flag>

Pre-requisites: None

Updates the entry with the specified dataset name in the ICCP.DLDATASET table.

Delete Dataset

Command: 5008

Data: <dataset name>

Pre-requisites: None

Deletes the entry with the specified dataset name in the ICCP.DLDATASET table.

12.1.4 DL Device Commands

Create Device

Command: 5009

Data: <device name>, <device type>, <device owner>, <domain reference name>, <device id>, <timeout>, <select before operate>, <checkback id>, <taggable>, <component alias>, <attribute name>, <attribute id>, <back indication>

Pre-requisites: None

Creates a new entry in the ICCP.DLDEVICE table.

Update Device

Command: 5010

Data: <device name>, <device type>, <device owner>, <domain reference name>, <device id>, <timeout>, <select before operate>, <checkback id>, <taggable>, <component alias>, <attribute name>, <attribute id>, <back indication>

Pre-requisites: None

Updates the entry with the specified device name in the ICCP.DLDEVICE table.

Delete Device

Command: 5011

Data: <device name>, <device owner>

Pre-requisites: None

Deletes the entry with the specified device name & device owner in the ICCP.DLDEVICE table.

12.1.5 DL Domain Commands

Create Domain

Command: 5012

Data: <domain name>, <bilateral table version>, <scope>, <server control center>, <client control center>

Pre-requisites: None

Creates a new entry in the ICCP.DLDOMAIN table.

Update Domain

Command: 5013

Data: <domain name>, <bilateral table version>, <scope>, <server control center>, <client control center>

Pre-requisites: None

Updates the entry with the specified device name in the ICCP.DLDOMAIN table.

Delete Domain

Command: 5014

Data: <domain name>

Pre-requisites: None

Deletes the entry with the specified domain name in the ICCP.DLDOMAIN table.

12.1.6 DL Point Commands

Create Point

Command: 5015

Data: <point name>, <point type>, <point owner>, <domain reference name>, <dl dataset>, <component alias>, <attribute name>, <attribute id>, <associated data>, <custom flag>, [manual source]

Pre-requisites: None

Creates a new entry in the ICCP.DLPOINT table. Manual source is optional. If it is not provided then the value will default to 0.

Update Point

Command: 5016

Data:<point name>, <point type>, <point owner>, <domain reference name>, <dl dataset>, <component alias>, <attribute name>, <attribute id>, <associated data>, <custom flag>, [manual source]

Pre-requisites: None

Updates the entry with the specified point name in the ICCP.DLPOINT table. Manual source is optional. If it is not provided then the value will default to 0.

Delete Point

Command: 5017

Data: <point name>, <point owner>

Pre-requisites: None

Deletes the entry with the specified point name & point owner in the ICCP.DLPOINT table.

If this point is configured as a circuit breaker in ARP_DLPOINT table, it will also delete that entry from that table.

If this point is configured as an ARP component in ARP_DLPOINT table, it will also clear the arp_dlpoint and arp_domainref fields from that table.

Update Point Dataset

Command: 5018

Data: <point name>, <point owner>, <dl dataset>

Pre-requisites: None

Updates the dataset entry of the point with the specified point name & point owner in the ICCP.DLPOINT table.

Delete Point Dataset

Command: 5019

Data: <dl dataset>

Pre-requisites: None

Clears the dl dataset field of all entries in ICCP.DLPOINT that match the specified dldataset.

12.1.7 DL Transferset Commands

Create Transferset

Command: 5020

Data: <transferset name>, <transferset role>, <start time>, <interval timeout>, <interval>, <grace period>, <time allowed to live>, <buffer time>, <integrity timeout>, <integrity check>, <critical>, <report by exception>, <event code requested>, <operator request>, <iccp priority>, <iccp routing>, <object change>, <other external event>, <enabled>, <domain reference name>, <dataset reference name>, <association reference>

Pre-requisites: None

Creates a new entry in the ICCP.DLTRANSFERSET table.

Update Transferset

Command: 5021

Data: <transferset name>, <transferset role>, <start time>, <interval timeout>, <interval>, <grace period>, <time allowed to live>, <buffer time>, <integrity timeout>, <integrity check>, <critical>, <report by exception>, <event code requested>, <operator request>, <iccp priority>, <iccp routing>, <object change>, <other external event>, <enabled>, <domain reference name>, <dataset reference name>, <association reference>

Pre-requisites: None

Updates the entry with the specified transferset name in the ICCP.DLTRANSFERSET table.

Delete Transferset

Command: 5022

Data: <transferset name>

Pre-requisites: None

Deletes the entry with the specified transferset name in the ICCP.DLTRANSFERSET table.

Delete Transferset Dataset Ref

Command: 5023

Data: <dataset name>

Pre-requisites: None

Clears the dataset field of all entries in ICCP.DLTRANSFERSET that match the specified dataset.

12.1.8 Doc Interpretation Incoming Commands

Create Doc Interpretation Incoming

Command: 5100

Data: <enmac process label>, <iccp type>, <operation>, <comments>

Pre-requisites: None

Creates a new entry in the ICCP.DOC_INTERPRETATION_INCOMING table.

Delete Doc Interpretation Incoming

Command: 5101

Data: <enmac process label>, <iccp type>, <operation>, <comments>

Pre-requisites: None

Deletes the specified entry from the ICCP.DOC_INTERPRETATION_INCOMING table.

12.1.9 Doc Interpretation Outgoing Commands

Create Doc Interpretation Outgoing

Command: 5102

Data: <enmac type>, <iccp type>, <comments>

Pre-requisites: None

Creates a new entry in the ICCP.DOC_INTERPRETATION_OUTGOING table.

Delete Doc Interpretation Outgoing

Command: 5103

Data: <enmac type>, <iccp type>, <comments>

Pre-requisites: None

Deletes the specified entry from the ICCP.DOC_INTERPRETATION_OUTGOING table.

12.1.10 ICCP Control Table Commands

Create ICCP Control Table

Command: 5104

Data: <component class>, <action>, <iccp control>, <is trigger>, <comments>

Pre-requisites: None

Creates a new entry in the ICCP.ICCP_CONTROL_TABLE table.

Delete ICCP Control Table

Command: 5105

Data: <component class>, <action>, <iccp control>, <is trigger>, <comments>

Pre-requisites: None

Deletes the specified entry from the ICCP.ICCP_CONTROL_TABLE table.

12.1.11 ICCP Interpretation Incoming Commands

Create ICCP Interpretation Incoming

Command: 5106

Data: <component class>, <attribute name>, <enmac value>, <enmac action>, <iccp value>, <meaning>

Pre-requisites: None

Creates a new entry in the ICCP.ICCP_INTERPRETATION_INCOMING table.

Delete ICCP Interpretation Incoming

Command: 5107

Data: <component class>, <attribute name>, <enmac value>, <enmac action>, <iccp value>, <meaning>

Pre-requisites: None

Deletes the specified entry from the ICCP.ICCP_INTERPRETATION_INCOMING table.

12.1.12 ICCP Interpretation Outgoing Commands

Create ICCP Interpretation Outgoing

Command: 5108

Data: <component class>, <attribute name>, <enmac value>, <iccp value>

Pre-requisites: None

Creates a new entry in the ICCP.ICCP_INTERPRETATION_OUTGOING table.

Delete ICCP Interpretation Outgoing

Command: 5109

Data: <component class>, <attribute name>, <enmac value>, <iccp value>

Pre-requisites: None

Deletes the specified entry from the ICCP.ICCP_INTERPRETATION_OUTGOING table.

12.1.13 ARP Dlpoint Commands

Create ARP Dlpoint

Command:5110

Data: <cb_dlpoint>, <cb_domainref>, <arp_simulated>, <arp_dlpoint>, <arp_domainref>, <arp_profile>

Pre-requisites: None

Creates a new entry in the ICCP.ARP_DLPOINT table.

arp_simulated

Values:

- 0 : ARP is not simulated
- 1: ARP is simulated

If ARP is simulated, arp_dlpoint and arp_domainref are left blank.

Delete ARP Dlpoint

Command: 5111

Data: <cb_dlpoint>, <cb_domainref>,

Pre-requisites: None

Deletes the specified entry from the ICCP. ARP_DLPOINT table.

12.1.14 ARP Profile Commands

Create ARP Profile

Command: 5112

Data: <profile_name>, <arp_timer>, <reclose_timer>, <number_shots>

Pre-requisites: None

Creates a new entry in the ICCP.ARP_PROFILE table.

Timer values are in seconds.

Delete ARP Profile

Command: 5113

Data: <profile_name>

Pre-requisites: None

Deletes the specified entry from the ICCP. ARP_ PROFILE table.

13. Other Instructions

Comment

Command: 999

Data: <comment>

These instructions are ignored and can be used to comment the PFL datafile.

Set Component Template

Command: 1005

Data: <template alias>

If the currently selected component does not have the requested attribute, and if the '-c' option flag has been used, normally, the attribute is copied from the component's template. This instruction allows a different component to act as the template—that is, it acts as an override template.

This must be specified after a component has been selected. Once set, this template alias is only valid for the currently selected component—if another component is selected, this override template alias is cleared.

End of File

Command: 0

Data: ENDSEC

This must be the last instruction to indicate the end of the file.

14. Example PFL Input Files

14.1 Example 1

As stated earlier, the Plant File Loader datafile is created from Command/Data pairs. The following example selects components and sets values for them.

```
999
Setting miscellaneous attribute values
999
Select Component using the ENMAC alias
1
ALIAS-858252-A
999
Populate 'Tele Status Colour' attribute
2
Tele Status Colour
3
83
999
Select Component using the Root path
1001
:EHV:11kv:Baillieston:Feeder 1:CB
999
Populate the 'Load Current Capacity' Attribute
2
Load Current Capacity
3
200A
999
Select Component using a unique attribute name/value combination
1006
31, Subtn Network Number, 567
Populate the 'Voltage' Attribute
2
Voltage
3
11kV
0
ENDSEC
```

In general the following format may be applied to the Plant File Loader datafile:

```
1
<alias>
2
<attribute name>
3

<attribute value>
```

Repeated for each attribute of the node

Repeated for each node

Where:

- `<alias>` is the alias of the next node as loaded in the transfer.
- `<attribute name>` is the name of the attribute to be set.
- `<attribute value>` is the value to which the attribute is to be set.

14.2 Example 2

The following example shows a datafile that would populate data in the RT card, scan /control point, and properties tables.

```
999
*** Select RTU
3000
TEST_RTU
999
 *** Create tagged input card 1 ***
3101
0,0,input card 1,input card tag 1
999
 *** Create tagged input card ***
3101
0,0,input card 1,input card tag 2
999
 *** Create tagged output card 1 ***
3101
1,0,output card 1,output card tag 1
999
 *** Create tagged output card 2 ***
3101
1,0,output card 2,output card tag 2
999
*** select input card 1 ***
3102
input card tag 1
999
*** Create scan point ***
3201
```

```
0,1,0,0,0,0,2,0,scan point 1, scan point 1 tag
999
*** select input card 2 ***
3102
input card tag 2
999
*** Create scan point ***
3201
0,1,1,0,0,0,2,0,scan point 2, scan point 2 tag
999
*** select  output card 1 ***
3102
output card tag 1
999
*** Create control point ***
3301
0,1,0,0,0,0,ctrl point 1, ctrl point 1 tag
999
*** select output card 2 ***
3102
output card tag 2
999
*** Create control point ***
3301
0,1,1,0,0,0,ctrl point 2, ctrl point 2 tag
999
*** Select property with id -1
3400
-1
999
*** Select scan point
3302
scan point 2 tag
999
*** Add property with value 10
3401
15
999
*** Select property with id -2
3400
-2
999
*** Select control point
3302
ctrl point 2 tag
999
*** Add property with value 20
3401
```

```
15
0
ENDSEC
```

14.3 Example 3

The following example illustrates how a datafile can be used to create components and attributes.

```
999
Creating components and attributes
999
Select Component, and if not found, attempt to create
1101
ALIAS-858252-A, ALIAS-873388-A, new component, ALIAS-234234-A
999
Build attribute commands
999
Set attribute Type for build (Table)
66
2
999
Set attribute Vector Size for build
64
10
999
Set attribute Table Size for build
65
5
999
Set attribute Column Names for build
8
ColumnName1, ColumnName2, ColumnName3, ColumnName4, ColumnName5
999
Set attribute Column DE Types for build
9
3, 2, 3, 4, 2
999
Select/Build the attribute
2
new attribute
0
ENDSEC
```

15. PFL Commands in Numerical Order

This table lists the commands in numerical order for quick reference.

Command	Description	Functional Area
0	End of File	Other Instructions
1	Select a Component by Alias	Selecting Components
2	Select an Attribute	Selecting Attributes
3	Store a Value	Populating Attributes
4	Store an Alias	Populating Components
5	Update Attribute Header	Populating Attributes
6	Store a Raw Value	Populating Attributes
7	Update a Component Header	Populating Components
8	Build Column Names	Building Attributes
9	Build Column DE Types	Building Attributes
60	Select a Column of a Table	Selecting Attributes
61	Select a Row of a Vector and a Table	Selecting Attributes
62	Store an Alarm Reference	Populating Attributes
63	Build DE Type	Building Attributes
64	Build Vector Size	Building Attributes
65	Build Table Size	Building Attributes
66	Build Attribute Type	Building Attributes
999	Comment	Other Instructions
1000	Store a Definition	Populating Attributes
1001	Select a Component by Relative Pathname	Selecting Components
1002	Select a Lookup	Populating Attributes
1003	Store Lookup in Value	Populating Attributes
1004	Store Lookup in Definition	Populating Attributes
1005	Set Component Template	Other Instructions

Command	Description	Functional Area
1006	Select a Component by Component Class and Specific Attribute	Selecting Components
1007	Select a Component and if not found, try an Alternative Component	Selecting Components
1008	Store a String for Later Use in a Value of Definition	Populating Attributes
1009	Specifying a Current Root Point	Selecting Components
1100	Select a Component by Alias Wildcard	Selecting Components
1101	Select a Component and if not found, try to Build a Component	Selecting/Building Components
1102	Update Attribute Table Size	Populating Attributes
3000	Select an RTU	RT Instructions
3001	Delete all Data	RT Instructions
3002	Place a Line under a FEP	RT Instructions
3003	Place an RTU under a Line	RT Instructions
3100	Create a Card	RT Instructions
3101	Create a Tagged Card	RT Instructions
3102	Select a Card by User Tag	RT Instructions
3103	Select a Card by ID	RT Instructions
3104	Update a Card	RT Instructions
3105	Delete a Card	RT Instructions
3200	Create a Scan Point	RT Instructions
3201	Create a Tagged Scan Point	RT Instructions
3202	Select a Scan Point by Tag	RT Instructions
3203	Select a Scan Point by ID	RT Instructions
3204	Select a Scan Point by Address	RT Instructions
3205	Update a Scan Point	RT Instructions
3206	Delete a Scan Row	RT Instructions
3300	Create a Control Point	RT Instructions

Command	Description	Functional Area
3301	Create a Tagged Control Point	RT Instructions
3302	Select a Control Point by Tag	RT Instructions
3303	Select a Control Point by ID	RT Instructions
3304	Select a Control Point by Address	RT Instructions
3305	Update Control Point	RT Instructions
3306	Delete a Control Row	RT Instructions
3400	Set a Property ID	RT Instructions
3401	Add a Value Property	RT Instructions
3402	Add a Reference Property	RT Instructions
3403	Update a Property	RT Instructions
3500	Add a Scan Reference	RT Instructions
3501	Delete a Scan Reference	RT Instructions
3600	Update a Scan Input Reference	RT Instructions
3700	Add a Control Definition	RT Instructions
3800	Set Default Card Type	RT Instructions
5000	Create Association	ICCP Instructions
5001	Update Association	ICCP Instructions
5002	Delete Association	ICCP Instructions
5003	Create Control Center	ICCP Instructions
5004	Update Control Center	ICCP Instructions
5005	Delete Control Center	ICCP Instructions
5006	Create Dataset	ICCP Instructions
5007	Update Dataset	ICCP Instructions
5008	Delete Dataset	ICCP Instructions
5009	Create Device	ICCP Instructions
5010	Update Device	ICCP Instructions
5011	Delete Device	ICCP Instructions

Command	Description	Functional Area
5012	Create Domain	ICCP Instructions
5013	Update Domain	ICCP Instructions
5014	Delete Domain	ICCP Instructions
5015	Create Point	ICCP Instructions
5016	Update Point	ICCP Instructions
5017	Delete Point	ICCP Instructions
5018	Update Point Dataset	ICCP Instructions
5019	Delete Point Dataset	ICCP Instructions
5020	Create Transferset	ICCP Instructions
5021	Update Transferset	ICCP Instructions
5022	Delete Transferset	ICCP Instructions
5023	Delete Transferset Dataset Ref	ICCP Instructions
5100	Create Doc Interpretation Incoming	ICCP Instructions
5101	Delete Doc Interpretation Incoming	ICCP Instructions
5102	Create Doc Interpretation Outgoing	ICCP Instructions
5013	Delete Doc Interpretation Outgoing	ICCP Instructions
5104	Create ICCP Control Table	ICCP Instructions
5105	Delete ICCP Control Table	ICCP Instructions
5106	Create ICCP Interpretation Incoming	ICCP Instructions
5107	Delete ICCP Interpretation Incoming	ICCP Instructions
5108	Create ICCP Interpretation Outgoing	ICCP Instructions
5109	Delete ICCP Interpretation Outgoing	ICCP Instructions
5110	Create ARP Dlpoint	ICCP Instructions
5111	Delete ARP Dlpoint	ICCP Instructions
5112	Create ARP Profile	ICCP Instructions
5113	Delete ARP Profile	ICCP Instructions

Table 1: PFL Commands in Numerical Order

Index

CTEs 25
Environment Requirements 6
ICCP Instructions 38
PFL
    Building Attributes 26
    Example Input Files 47
    File Format 12
    Other Instructions 46
    Populating Attributes 21
    Populating Components 18
    RT Instructions 28
    Selecting Attributes 20
    Selecting Components 13
    Selecting/Building Components 16
PFL Command Line Parameters 7
PFL Commands in Numerical Order 51
PFL Error Reporting 11
Plant File Loader (PFL) Utility 6