–METASYS®

DDL Programmer's Manual

Introduction 3
Preface 3
Before You Begin 3
About this Manual 3
Overview 4
Workstation File 7
Global File 8
Model File 9
NC File 9
When DDL is Compiled 11
Interaction with Other Tools 14
CAE 14
GPL 14
Online Generation and Upload 14
JC-BASIC 14
Hardware Requirements 15
Source File Development 3
Overview 3
Syntax 4
File Keywords (Headers) 4
Comment Lines 5
Blank Lines 5
Main Keywords 5
Subkeywords 8
Parameters 9
Optional Parameters 13
Conditional Parameters 13
Other Format Considerations 14
Semantic Rules 16
Example 17
Compiler 1
Overview 1
How the DDL Compiler Works 2
Demand Limiting/Load Rolling Databases 2
What the DDL Compiler Does Not Do 2
Adding and Deleting Objects 3
Incremental vs. Full Compile 5
Full Compile 5
How a *Full* Compile Affects Demand Limiting/Load Rolling Databases 5

© January Johnson Controls, Inc.

Incremental Compile 6
Incremental Compile Considerations 6
Error Handling 7
Errors and Warnings 7
Locating Errors and Warnings 7
Software-to-Software References 8
Software-to-Hardware References 9
CS Object-to-Software Model References 10
Feature-to-Object References 10
Software-to-Feature References 10
Duplicate Objects 10
Duplicate Addresses 10
Conditionals 11
References Within Global Files 11
BEFORE Executing the Compiler 12
Setting the FMSDOS Environment Variable 12
Setting WIN.INI File Variables 13
Make Sure All Directories Exist 15
Executing the Compiler 16
Database Output 17
Screen Interaction 18
Compiler List File 19
Downloading and Uploading 19
Operator Workstation Disk Layout for DDL 22
FMS System Software 22
Models Archive 25
Database 25
FMS Archive Databases 25
FMSDOS Environment Variable 25
Fixed METASYS\SB4W Directory 26
WIN.INI File 28
FMSData Files 30
Reference 1
Description of File Tables 1
Column Headings 1
Workstation Network/Port Configuration File Syntax 4
Workstation File @NET Keyword 4
Workstation File NET Keyword 5
Workstation File Port Keyword 7
Global File Syntax 8
Global File @GLOBAL Keyword 8
Global File DEFDES Keyword 9
Global File GRP (PC Group) Keyword 10
Global File NC Keyword 11
Global File PC Keyword 14
Global File PTR Keyword 17
Global File RPT Keyword 19

Global File SYS Keyword 21
Model File Syntax 22
Model File @MODEL Keyword 22
Model File CSMODEL Keyword 23
NC File Syntax 31
NC File @NC Keyword 31
NC File ACM Keyword 32
NC File AD Keyword 37
NC File AI Keyword 40
NC File AOD Keyword 51
NC File AOS Keyword 53
NC File BD Keyword 58
NC File BI Keyword 61
NC File BO Keyword 67
NC File C210A Keyword 73
NC File C260A Keyword 75
NC File CARD Keyword 77
NC File CS Keyword 79
NC File D600 Keyword 81
NC File DCDR Keyword 83
NC File DCM140 Keyword 85
NC File DCM Keyword 86
NC File DELCARD Keyword 87
NC File DELETE Keyword 88
NC File DELSUBORDINATE Keyword 89
NC File DELTZ Keyword 89
NC File DLLR Keyword 90
NC File DSC Keyword 93
NC File DSC8500 Keyword 94
NC File FIRE Keyword 95
NC File FPU Keyword 97
NC File JCB Keyword 98
NC File LCD Keyword 99
NC File LCG Keyword 100
NC File LON Keyword 104
NC File MC Keyword 106
NC File MSD Keyword 110
NC File MSI Keyword 114
NC File MSO Keyword 120
NC File N2OPEN Keyword 129
NC File PIDL Keyword 130
NC File READER Keyword 135
NC File SUBORDINATE Keyword 138
NC File TIMEZONE Keyword 150
NC File XM Keyword 153
NC File ZONE Keyword 155
Decompiler 1

Overview 1
Purpose 1
Input to the Decompiler 1
Output From the Decompiler 2
How the Decompiler Works 2
Guidelines For Effective Operation 3
Executing the Decompiler 3
Command Line 4
Overwriting Files 5
Procedure 6
Screen Output 7
Error Notification 8
Where Decompiled Files Go 8
What Decompiled Files Look Like 9
Example Workstation Network/Port File 10
Decompiler Summaries 11
Error Handling 14
Error Examples 15
NC Database Decompiling 16
Differences Between Decompiler and Compiler 17
Recompiling 17
Compatibility With Previous Release Databases 18
Advanced Topics 3
Application Questions 3
Changing an NCM Node or Subnet Address 3
Moving an NCM to a Different Network (Network Name Changes) 4
Changing an NCM Name 6
Changing a Network Name 7
Recompiling NC Files After Compiling Global or Model Files 7
Retranslating GPL Files after Compiling NC Files 8
NET File for OWS with Multiple Networks 9
Configuring DDL Files for the M5 Workstation ... 10
NET File Keyword Order 10
What is Going on During a DDL Compile 11
Error and Warning Messages 12
ERRORLOG.TXT File Messages 15

Introduction

Preface

Data Definition Language (DDL) is the programming language you use to create a facility database.

Before You Begin

Before using the DDL feature, you must be familiar with the elements of a Metasys® Network, as described in the *Metasys Network Technical Manual*. In particular, review the characteristics of hardware and software objects. For details on the system engineering strategy, refer to the *Engineering Guide* in the *Metasys Network Technical Manual*.

To use the DDL compiler and DDL decompiler, you must have basic personal computer skills and an understanding of DOS concepts, such as how to create directories and files.

To load DDL, refer to the documentation that accompanies the software.

About this Manual

The *DDL Programmer's Manual* describes how to create, edit, and compile DDL source files, and decompile the archive database. It divides into these sections:

Introduction describes the purpose and organization of DDL, and an overview of the steps to create and implement DDL files. The relationship of DDL to other software tools is explained, as are the hardware requirements.

Source File Development contains detailed instructions for developing DDL source files.

Compiler describes how to use the compiler and interpret the messages it outputs.

Reference is a detailed description of the DDL syntax and semantic rules. It includes examples.

Decompiler describes how to use the decompiler and how decompiled files appear.

Advanced Topics contains answers to some commonly asked questions regarding DDL.

Overview

Data Definition Language (DDL) is a text-based program used to identify the components on a Metasys Network, organize how these components connect to each other, and attach the various hardware and software objects to the appropriate system.

DDL establishes the source files for hardware and object databases, and is therefore the first step in configuring the system software.

With DDL, you use any ASCII editor (DFEDIT is provided) to create four types of source files:

- Workstation Network/Port File
- Global File
- Model File
- NC File

The DDL compiler translates the DDL source files into an archive database on the disk of the Operator Workstation (OWS), and an Operator Workstation specific database.

The archive database can be uploaded (NC or Global upload), in which case it is a copy of the online database (residing in the NC or OWS). The archive database consists of:

- *global data*, which can be downloaded to the online global databases of the Operator Workstation.
- *NC data*, which can be downloaded to an NCM and its associated hardware devices.

The OWS-specific database is directly accessed online, and is the *only* copy of the OWS-specific information. (The one copy functions as both archive and online versions.) The OWS-specific database consists of:

- *OWS configuration data*, which defines the networks the OWS can access, and the ports the access can occur on.
- *OWS-specific data*, which defines software model data known only to a single OWS, and PC group information, which defines the Network Map of the OWS.

Alternatively, the configuring process, or additions and corrections to the database generated by DDL, can be accomplished with the Metasys FMS *Online* Generation program, then uploaded to synchronize the archive database files. Figure 1 illustrates DDL's role in the process.

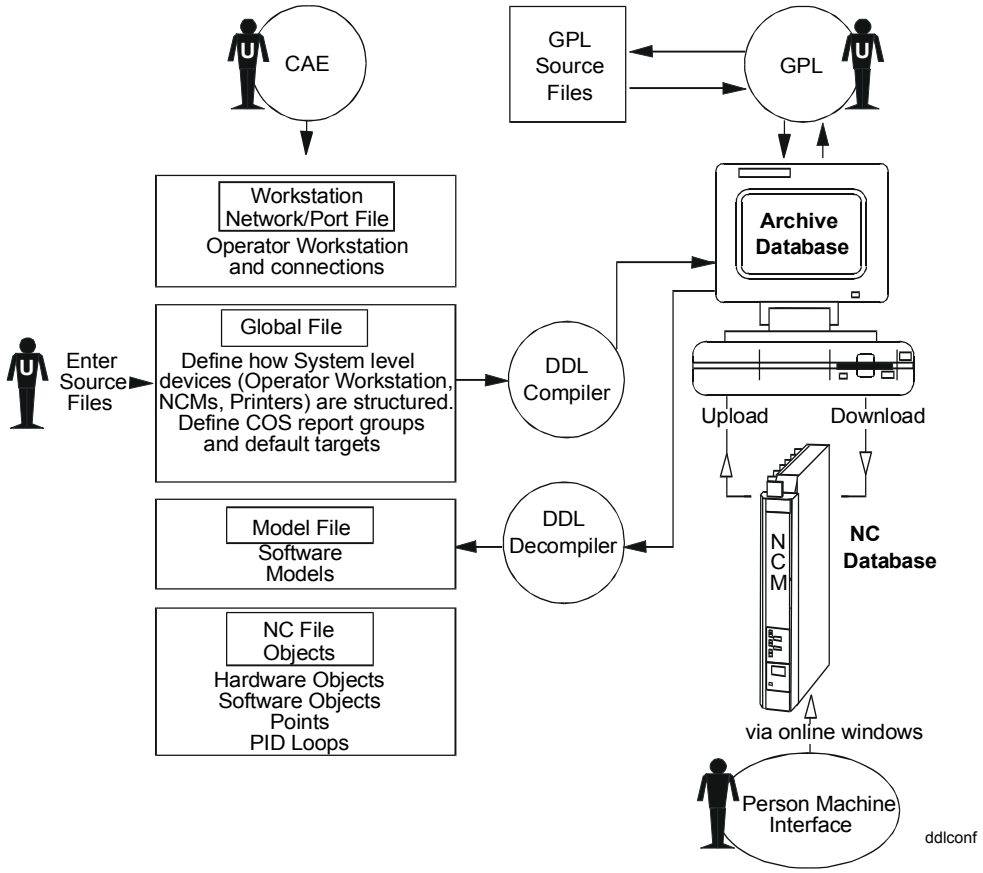


Figure 1: DDL in the Network Configuration Process

The only feature databases DDL generates are TIMEZONE and CARD databases for Access, Load Groups and Loads for Demand Limiting/Load Rolling, and Event Scheduling for Lighting.

DDL will not generate Graphic Programming Language (GPL) source files. However, when the CAE system generates DDL source files, special source files with DDL-like keywords are generated, which can create GPL source files when compiled by a special DDL/GPL compiler. After the DDL files are compiled, GPL can add, delete, or modify software objects and process objects.

Workstation File

Each **Workstation Network/Port File** (called the "Workstation File") defines the networks to which that Operator Workstation has access, and describes how the workstation connects (over the N1, Dial-up, or directly into the NCM) to each network. (See Figure 2).

The Port configuration assigns port connections between the Operator Workstation and equipment such as a printer, modem, or mouse. If necessary, the baud rate can also be configured.

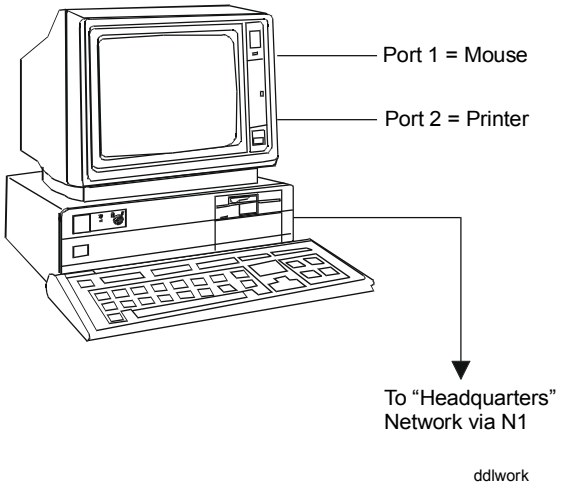


Figure 2: What the Workstation Network/Port File Defines

The Workstation Files must be compiled first on each workstation. Each Operator Workstation requires a unique file.

Global File

The second file to compile is the **Global File**. There is one Global File for each network defined in the Workstation (@NET) File. A Global File defines:

- *Devices*--Operator Workstations, NCMs, and printers
- *Systems*--associating objects to specific NCMs (conversely, one NCM can contain more than one system)
- *PC Groups*--independently set up at each Operator Workstation to group systems into larger categories
- *Report Groups*--classifies object reports and default destinations of those reports for the network

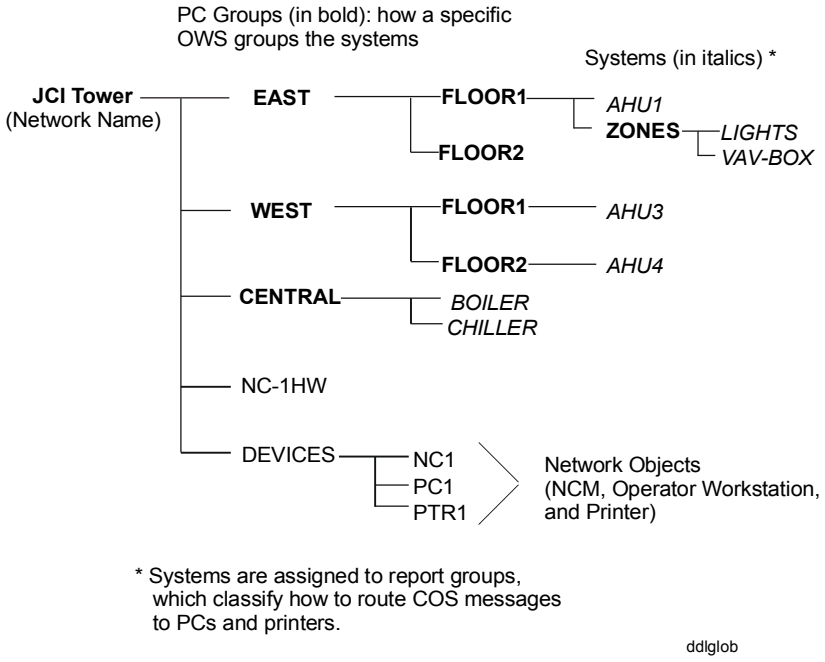


Figure 3: What the Global File Defines

The global database defined by a Global File, with the exception of the PC Group hierarchy, is shared between all Operator Workstations and NCMs on the network.

PC Groups

The same Global File may reside on each Operator Workstation in the network. If the PC Group structure varies between workstations on the network, then different Global Files reside on the workstations, reflecting the individual PC Group structures.

Note: To guarantee that the PC Group hierarchy of each workstation is correct, compile the Global File for a particular workstation on that workstation before tying it into the N1 network.

Model File

The **Model File** contains the database of software models. These software models provide default values used when defining generic Control System (CS) objects. Generic Control System objects are software representations of application specific controllers (AHU, LCP, UNT, VAV, VMA, PHX, MIG, VND).

The Model File must be compiled before the NC File containing CS objects that reference models. However, the Model File can be compiled before or after Workstation and Global Files.

NC File

The **NC File** generates a database of hardware and software objects, Demand Limiting/Load Rolling objects, and Access features (TIMEZONE and CARD) for a particular NCM. The figure on the next page illustrates one NC File's database.

The NC Files are fourth in order to compile. A unique NC database resides in each NCM, corresponding to the archive database held in the Operator Workstation. (An Operator Workstation can hold any number of NC databases; however, there is no requirement that one of the workstations hold *all* the NC databases.)

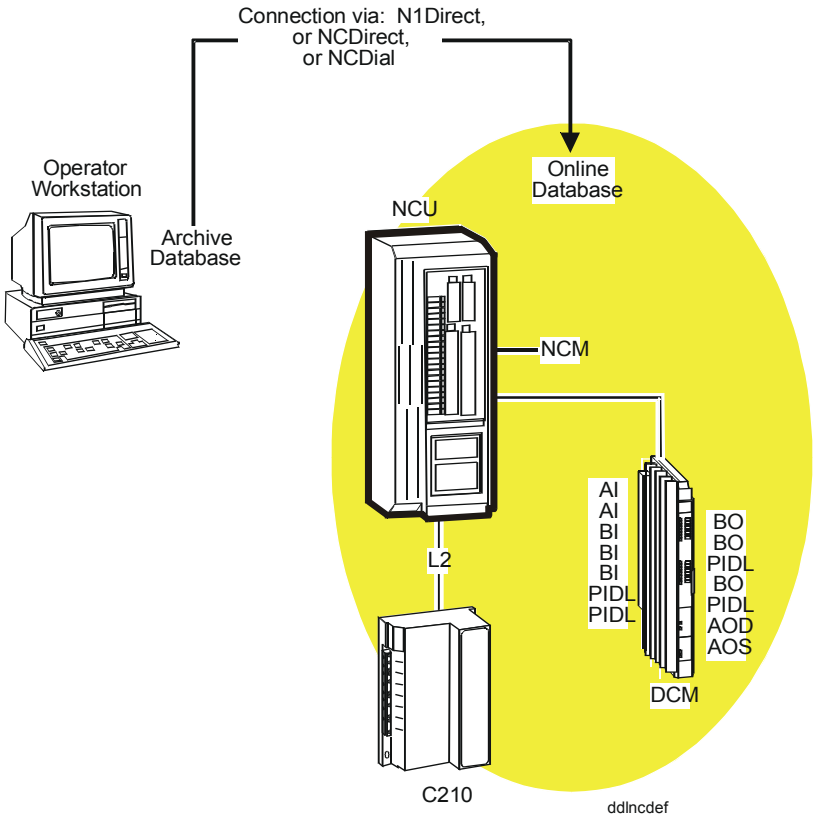


Figure 4: What the NC File Defines

Listed below are the objects and Access features for which the NC File generates archive database records.

- Analog objects (ACM, AD, AI, AOD, AOS)
- Binary objects (BD, BI, BO)
- Control System objects (C210A, C260A)
- Generic Control System objects (CS)
- Multistate objects (MSD, MSI, MSO)
- Multiple Command objects (MC)
- Subordinates associated with MC object
- Lighting Control Groups (LCG)
- Lighting Control Device hardware objects (LCD)
- PID Loops (PIDL)
- Fire software objects (ZONE)

- Access software objects (READER)
- Fire controller hardware objects (FIRE)
- Access controller hardware objects (D600)
- DCDR hardware objects (LCP, DX9100, DX91ECH, DC9100, DR9100, TC9100, XT9100, XTM)
- LON hardware objects (e.g., LONTCU)
- Digital Control Module hardware objects (DCM and DCM140)
- Expansion Module hardware objects (XM)
- N2OPEN hardware objects (AHU, MIG, NDM, PHX, UNT, VAV, VMA, VND)
- L2 hardware objects (DSC)
- S2 Migration hardware objects (DSC8500, FPU)
- Control processes (JCB, directory entry only)
- Demand Limiting/Load Rolling objects (DLLR)
- Access features (TIMEZONE and CARD)

When DDL is Compiled

You must compile the Workstation Network/Port, Global, and NC files in that specific order, since each file builds upon the previous file's data. The Model file must be compiled before the NC files that contain CS objects. However, the Model file can be compiled before or after Workstation or Global files.

The DDL compiler generates an archive database and an Operator Workstation-specific database from the DDL source files. Part of these databases is used by the Operator Workstation features, part for the global databases for all network devices (Operator Workstations, NCMs); with the rest downloaded to the NCMs and their associated devices. The upload and download features synchronize changed archive and operational databases.

After compiling the DDL files, there are four places where the information resides, as shown in Figure 5:

- *DDL File*--the original, ASCII text source file and the compiler list file (.dlt).
- *Archive Database*--residing on the Operator Workstation. This consists of downloadable NC and global database files, created by the DDL compiler. It may contain modifications that have been added by online generation and uploaded. GPL may also add to, delete from, or modify this database.

- *Operational Database*--residing at the NCM in the case of the downloaded NC database files, and on the workstation, in the case of the OWS global files. These are working copies of the archive database plus any modifications that have been entered online, whether or not they have been uploaded to the archive database residing on the Operator Workstation.
- *OWS-specific Database*--residing on the OWS. Contains the *only* copy of the OWS-specific information. (The one copy functions as both archive and online versions.) The OWS-specific database consists of: OWS configuration data, which defines the networks the OWS can access, and the ports the access can occur on; software model data known only to a single OWS; and PC group information, which defines the Network Map of the OWS.

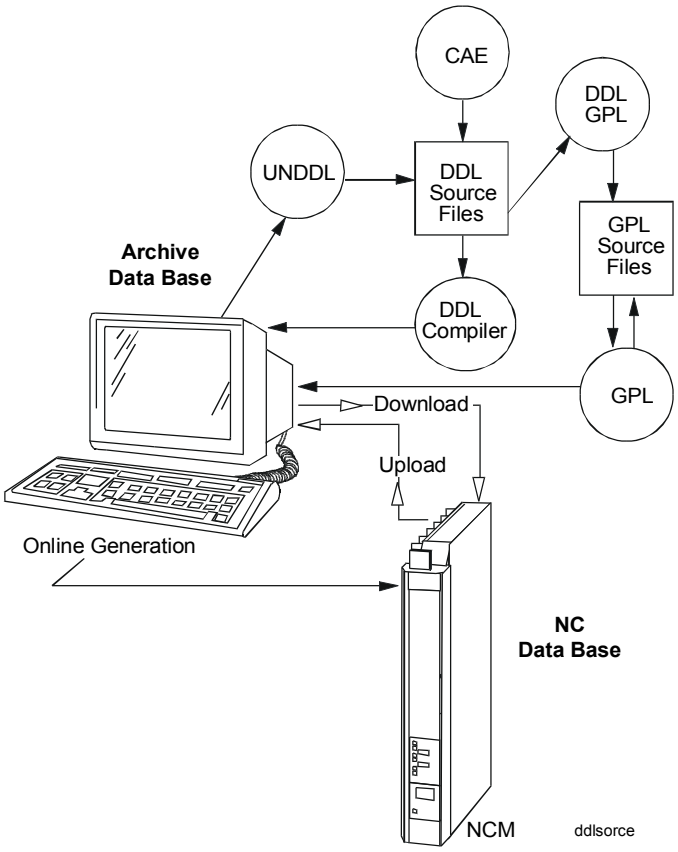


Figure 5: DDL Source Files, Archive Database, and Operational NC Database

Two kinds of compilations can be invoked: Full or Incremental.

- A *full compile* deletes and fully replaces any existing database which DDL can create. That is, for Net and Model files, the entire existing OWS-specific database is deleted and replaced by a new database generated from the DDL source. For a Global file, the entire existing global archive database is deleted and replaced by a new database generated from the DDL source. For a NC file, all portions of the existing NC archive database which DDL can create (all object data and Access features), and JC-BASIC (GPL) process object code, are deleted. The NC file source code only replaces object and Access feature data in the database.

  Use the full compile to create the original archive and OWS-specific databases, or when making changes so numerous as to warrant replacing the entire database.

- An *incremental compile* adds items to the archive and OWS-specific databases and, in the case of the NC+ file, deletes items from the NC archive database. Items that were previously added remain intact. “Deletes” may only be performed when incrementally compiling an NC file.

  To avoid destroying previous object changes that were uploaded from the operational database, use *only* incremental compiles when making changes to the archive database.

Since both full and incremental NC compiles do *not* delete any Scheduling, Trend, and Totalization data from the archive database, caution should be exercised because an object may no longer exist in the archive database, but a feature may continue to reference the object.

Refer to the *Compiler* section for complete details on using the compiler.

Interaction with Other Tools

Described below are the means by which other Metasys software tools interact with DDL: CAE, GPL, Online Generation, and JC-BASIC.

CAE

- Composes DDL source files.

  As the CAE tool outlines the systems and equipment for the facility, it also composes DDL source files. Inside these files are keywords to generate the Workstation, Global, Model, and NC databases.

GPL

- GPL can modify software objects in the archive database.

  When using GPL, a user can add, modify, or delete software objects from the archive database. For example, while in the GPL Editor, a user can manipulate the archive database by adding or deleting a PID Loop, or by changing its parameters.

  For further information, refer to the *GPL Programmer's Manual.*

Online Generation and Upload

- An uploaded operational database overwrites its archive database.

  Online generation modifies the contents of the operational database. Changes there may be uploaded to write over the archive database. As a result, both DDL and Online generation (uploaded) can change the archive database.

  In the case of the OWS-specific database (which consists of OWS configuration data, Model data, and PC group information), online generation can access and modify the database directly (since one copy functions as both archive and online versions).

JC-BASIC

- JC-BASIC processes do *not* remain intact.

  JC-BASIC processes are consistent with other DDL objects. Therefore, all JCB processes (including process object code) are deleted at full compiles. All JC-BASIC processes can also be deleted via the DELETE keyword.

Hardware Requirements

The DDL compiler can operate on any PC or portable PC that meets the same requirements as the basic Operator Workstation. (It is possible to build the source file with any editor on any PC, then bring it in to compile on a workstation.) Refer to the *Metasys Network Technical Manual* for details.

Before calling DDL, close the Windows® software.

The DDL compiler cannot be executed from the Network Terminal (NT).

Source File Development

Overview

The DDL language is defined by an explicit language syntax. This section discusses the general rules, usage, and format of the language. For specific details regarding syntax elements, see the *Reference* section of this manual.

When creating or editing a DDL source file:

- Avoid all control characters (characters formed by simultaneously holding down the Control key with another key).
- Use DDL as the extension of the file name.

You may use any ASCII text editor to create DDL source files. (DFEDIT is provided with the Operator Workstation.)

The format for a source file begins with a file keyword, and is followed by object definitions composed of main keywords and subkeywords.

File Keywords (headers) describe the type of file (e.g., "@GLOBAL") that follows.

Main Keywords, alone or with *Subkeywords*, specify the object being defined and determine the database location to which you are writing information (e.g., "XM" writes a Point Multiplex Module record in the NC archive database).

A line of code is a keyword followed by its parameters, if any. Parameter data is the information actually stored; the information it accepts is configured by its *Type* (e.g., integer or character string), *Restrictions, Defaults,* and any appropriate notes.

Each syntax element is described on the following pages. Format considerations (e.g., "alphabetic characters are not case sensitive") and Semantic Rules (e.g., "the low-limit value must be less than the high-limit value") are then explained, followed by a typical source file, with commentary.

Syntax

File Keywords (Headers)

Each type of file (Workstation, Global, Model, NC) must be a separate DOS file. To indicate the type of file you are creating (compiling), specify the File Keyword on the first non-comment line in the file. An amendment to the keyword also indicates whether it compiles fully or incrementally.

- Workstation Network/Port File

  The header for the Workstation Network/Port File is:

  @NET

  To indicate an incremental compile, add a "+" to the file keyword:

  @NET +

  Example: ***`@NET`***

WARNING: If you intend to incrementally compile, ensure that the file keyword contains a "+" (for "incremental" compile) before running the compile. Otherwise you will run a full compile and erase any uploaded object changes to the archive databases.

- Global File

  The header for the Global File is:

  @GLOBAL networkname

  Incremental compiles require a "+":

  @GLOBAL + networkname

  Example: ***`@GLOBAL "JOHNSON"`***

  The "network name" parameter (1 - 8 characters, enclosed in double quotes) identifies the network to which this file applies, and is the same as that used in the Workstation Network/Port file (NET) keyword.

  The global database for the network is generated when this file is compiled.

- Model File

  The header for the Model File is:

  @MODEL

  Incremental compiles require a "+":

  @MODEL +

  Example: `@MODEL`

- NC File

  The header for the NC File is:

  @NC network name, NCMname

  Again, incremental compiles require a "+":

  @NC + network name, NCMname

  Example: `@NC "JOHNSON", "LABNC"`

  The "network name" parameter (1 - 8 characters, enclosed in double quotes) identifies the network to which this file applies, and is the same as that used in the Global File for the network.

  The "NC name" parameter (1 - 8 characters, enclosed in double quotes) identifies the name of the NCM itself to which the database will be downloaded. The NCM name is the same as that used in the NC Definition keyword (NC) in the Global File.

Comment Lines

A comment line is designated by an asterisk (*) as the first non-blank character of the line. The compiler ignores all comment lines.

Blank Lines

A blank line is a line in the source file with no characters other than blanks. The compiler ignores all blank lines.

Main Keywords

Main Keywords are fixed ASCII strings (mnemonics) that designate a specific item or object in the database that the file can generate. Each item or object that the DDL compiler generates has a keyword.

Examples of main keywords are PORT (to supply data on what devices are attached to which ports of this Operator Workstation), and ACM (to generate an ACM object). The keyword must be the first item on a line of DDL (but may be preceded by one or more blanks). Following the keyword, on the same input line, there may be parameters that define the specific data to become part of the database records. Table 1 lists all the valid Main Keywords for DDL.

Note: The main keywords in this table and in the Reference section of this manual are listed in alphabetical order within their file type. There is no correlation between the order of keywords in this table and the order the keywords need to appear in the file.

File	Keyword	Result
Workstation Network/Port Configuration File @NET	NET	Defines a network accessible from this Operator Workstation, (via N1-direct, NC-direct, or NC-dial) and adds its name to the network map.
	PORT	Configures one of the ports on this Operator Workstation.
Global File @GLOBAL	DEFDES	Defines a default device.
	GRP	Defines a PC group and its parent.
	NC	Defines an NCM.
	PC	Defines an Operator Workstation, either N1-direct, NC-dial, or NC-direct.
	PTR	Defines a printer, either PC-direct, NC-direct, or NC-dial.
	RPT	Defines a COS (Change-of-State) report group and its targets.
	SYS	Defines a system name and optionally assigns it to a PC group.
Model File @MODEL	CSMODEL	Defines a software model.
NC File @NC	ACM	Defines the Accumulator object.
	AD	Defines the Analog Data object.
	AI	Defines the Analog Input object.
	AOD	Defines the Analog Output Digital object.
	AOS	Defines the Analog Output Setpoint object.
	BD	Defines the Binary Data object.
	BI	Defines the Binary Input object.
	BO	Defines the Binary Output object.
	C210A	Defines the Control System object for a C210A controller.
	C260A	Defines the Control System object for a C260A controller.

Continued on next page . . .

File (Cont.)	Keyword	Result
NC File @NC (Cont.)	C260X	Defines the Control System object for a C260X controller.
	C500X	Defines the Control System object for a C500X controller.
	CARD	Defines the CARD Access feature.
	CS	Defines the Generic Control System object.
	D600	Defines the D600 controller hardware object.
	DCDR	Defines the LCP, DX9100, DX91ECH, DC9100, DR9100, TC9100, XT9100, or XTM hardware object.
	DCM	Defines the DCM controller hardware object.
	DCM140	Defines the DCM140 controller hardware object.
	DEL SUBORDINATE	Deletes a subordinate from an MCO.
	DELCARD	Deletes the CARD Access feature.
	DELETE	Allows the deletion of an object.
	DELTZ	Deletes the TIMEZONE Access feature.
	DLLR	Defines the Load Group Object.
	DSC	Defines the C210A or C260A hardware object.
	DSC8500	Defines the DSC8500 controller hardware object.
	FIRE	Defines the FIRE controller hardware object.
	FPU	Defines the FPU controller hardware object.
	JCB	Adds a process name only (use GPL or JC-BASIC to create the process object).
	LCD	Defines the lighting controller hardware object.
	LCG	Defines the Lighting Controller Group object.
	LON	Defines the LON device hardware object (LONTCU)
	MC	Defines the Multiple Command object.
	MSD	Defines the Multistate Data object.
	MSI	Defines the Multistate Input object.
	MSO	Defines the Multistate Output object.
	N2OPEN	Defines the AHU, MIG, NDM, PHX, UNT, VAV, VMA, or VND controller hardware object.
	PIDL	Defines the PID Loop object for a DCM.
	READER	Defines the READER software object for a D600.
	SUBORDINATE	Defines a subordinate for the MC object.
	TIMEZONE	Defines the TIMEZONE Access feature.
	XM	Defines the point multiplex module hardware object.
	ZONE	Defines the ZONE software object for a FIRE controller.

Table 1: Main Keywords in Workstation, Global, Model, and NC Source Files

The following example shows how the NET (network) keyword might be used to describe a network called JOHNSON with an expanded ID of MICHIGAN ST and an Operator Workstation name of DOWNTOWN. The N1DIRECT subkeyword follows it to define how the Operator Workstation is connected in the network.

Example:

```
NET "JOHNSON", "MICHIGAN ST", "DOWNTOWN"
N1DIRECT 1, 100
```

Subkeywords

Since data for an item will not always conveniently fit on a source line with the Main keyword, DDL uses subkeywords to define the rest of the data for the item or object. In that way, the Main keyword line, together with all of the subkeywords, completely defines the object.

Some subkeywords are optional. When defining an item and omitting the optional subkeyword, the compiler sets all of the fields defined by that subkeyword to designated default values.

In the previous example, a subkeyword followed the NET main keyword. The subkeyword, N1DIRECT, selects the connection type for that network. The parameters for N1DIRECT specify subnet address 1 and node address 100. (The blanks preceding N1DIRECT in the example are optional. They are used here only to demonstrate a method that visually distinguishes a subkeyword from the Main keyword.)

Parameters

Each parameter defines an actual value, which is inserted into the database record for the item being generated.

Valid parameters, if any, appear after a Main keyword or subkeyword, and are separated from the keyword by one or more spaces. Commas separate the parameter fields.

Example:

```
NET "JOHNSON", "MICHIGAN ST", "DOWNTOWN"
```

Each parameter has a specific data type. The parameter types are:

Boolean Y/N (Y=Yes, N=No)

Example: `Y`

Boolean[n] Allows up to "n" integers in brackets ([]) to set the Boolean for the corresponding items to "Y" in the array.

The following example indicates that the first, twelfth, sixteenth, and thirty-ninth values in the array are to be Yes. All others are No.

Example: `[1,12,16,39]`

Do not repeat numbers within the brackets. For example, the following will result in an error, because the 3 is repeated: `[1,3,5,7,3]`

chars(n) A character string that can contain up to "n" characters. Enclose all strings in quotation marks (" ").

Example: `"EASTWING"`

To have a double quotes (") as a character in this parameter, type "" (two double quotes, no space). This counts as only one character.

Example: `" "" of WG"`

DOSfile A valid DOS file name (without an extension) is enclosed in quotation marks (" ") and contains up to eight valid DOS characters. Restricted characters are:

period (.)

double quotation marks (" ")

slash (/)

back-slash (\)

square brackets ([])

colon (:)

vertical bar (|)

angle brackets (<>)

plus sign (+)

equals sign (=)

semicolon (;)

comma (,)

single quotation marks (' ')

asterisk (*)

question mark (?)

space ( )

and characters whose ASCII codes are less than Hex 20 (Hex 0 through Hex 1F)

Although not restricted, do not use the following characters when defining points that may be accessed by the JC/85 Gateway. The terminal on the JC/85 does not have the capability to type them in.

braces ({})

tilde (~)

The underscore character is allowed in names; however, GPL cannot display it. Therefore, it will not appear on the screen, and where the underscore is used, the character will appear to be a blank.

Finally, the following names are restricted:

AUX	CON	LPT3
AUXIN	COS	MODELS
CAL1	DDL	NUL
CLOCK$	DEVICES	PRN
COM1	GPL	PROCESS
COM2	INSTRUCT	SB4W
COM3	LPT1	SYSTEMS
COM4	LPT2	

Example: `"FIRECRIT"`

fp Floating point (number using a decimal point or scientific notation). Whole numbers, however, require no decimal point (e.g., 55). If no restrictions are specified in the *Reference* section, its range is -3.4E38 to +3.4E38.

Example: `62.5`

integer A positive or negative whole number, or zero.

If no restrictions are specified in the *Reference* section, its range is -32768 to +32767.

Example: `12`

long A positive or negative whole number, or zero.

If no restrictions are specified in the *Reference* section, its range is -999999999 to +999999999.

Example: `6767898`

name An item name enclosed in quotation marks (" "). It names FMS items such as systems, objects, and networks. It consists of a string of one to eight characters.

Characters use the same restrictions as DOS file parameters, since they are often used to name DOS subdirectories.

Example: `"AHU1"`

phone#(n) A string that can contain up to "n" characters, enclosed in quotation marks (" "). It can consist of the following characters:

0 to 9

, (comma)

( ) (left and right parentheses)

- (minus sign)

(pound sign)

* (asterisk)

Example: `"(414)555-4980, #4580"`

report-type Defines the type of report issued, by using one of the following strings enclosed in quotation marks:

status, follow-up, critical, trans, history, trend, total, card reader

Example: `"critical"`

Other restrictions for specific parameter values appear in the *Reference* section with each parameter.

Optional Parameters

Some parameters are optional. You may omit an optional parameter from the input line.

- Specify two commas in a row to indicate which optional parameter you are omitting.
- To omit optional parameters at the end of the line, omit the parameter(s) and any trailing comma(s).

If you omit an optional parameter, the compiler uses the default value. (If you omit an optional subkeyword, the compiler uses the same indicated default values.)

In the following example, the compiler would use a null (blank) description, because [null] appears as the default value for description in the *Reference* section.

Example:

```
NET "JOHNSON",,"DOWNTOWN"
```

Conditional Parameters

Some parameters are conditional. Only use a conditional parameter when the value of some other parameter requires it. For example, for an ACM object residing on an XM, the debounce filter is only valid if the point is not a “Form C” type. In other words, using the debounce filter is conditional on the type selected. Specified conditions appear in the *Reference* section.

Conditional parameters can be either optional or required:

- Optional Conditional--For an Optional Conditional parameter, you do not need to enter the value even if the conditions are met since a default value will be used.
- Required Conditional--For a Required Conditional parameter, you must enter the value when the conditions are met.

For example, for a PID Loop, an input can be either a Value or a Point Reference. If you choose Value, you must choose to not specify the system and object Name parameters. If you choose Point Reference, then you must specify the Name parameters for the referenced point.

Other Format Considerations

The following rules govern the layout of the source lines that the compiler will accept:

- The source input is line oriented, with each keyword or subkeyword beginning on a new logical line.
- The File Keyword (@xxx) must be the first non-blank, non-comment line in the file.
- The Main Keyword or subkeyword must be the first non-blank item on the logical line.
- The Main Keyword or subkeyword must be separated from the first parameter (if any) by at least one space.
- Parameters are separated by commas. Optional parameters that are omitted are indicated by adjacent commas. Optional parameters at the end of the line do not require trailing commas.
- The compiler recognizes only ASCII characters. It allows no control characters or tabs.
- A physical line is limited to 132 characters and is ended by a carriage return and/or line feed.
- A logical line consists of a keyword or subkeyword and all of its parameters. It is contained either on one physical line or several physical lines linked by a continuation character (a backslash). There is no limit to the number of characters on a logical line.
- The continuation character is a backslash (\). It must occur as the last non-blank character on the line, normally before or after a comma separating two parameters. If used before a comma, the comma goes on the next line. You may also use the backslash between a keyword (or subkeyword) and its first parameter.

Example, using the backslash after a comma:

```
NET "JOHNSON","MICHIGAN-STREET",\
"DOWNTOWN"
```

Example, using the backslash before a comma:

```
NET "JOHNSON","MICHIGAN-STREET"\
,"DOWNTOWN"
```

You may not use the backslash as a continuation character within a parameter, except within the Boolean[n] parameter, between integers.

Example:

```
OUTPUTS  [1,5,9,12,25,29],[1,5,9,\
12,25,29]
```

- A single parameter may not cross a physical line (the Boolean parameter is an exception). This means the longest string, including its double quotes, is 132 characters.
- Blank lines and comment lines may occur anywhere in the source file except between two continuation lines.
- An asterisk (*) must begin a comment line as the first non-blank character of the line. Comment lines may not be continued; if you desire multiple comment lines, simply begin another line with an asterisk.
- Blanks are allowed anywhere on a line except within the characters of a single keyword or parameter (unless the blank occurs in a string inside double quotation marks).

  Blanks are required whenever you must separate two items on a line, other than two parameters. For example, you must use a blank space between a keyword (or subkeyword) and its first parameter.

  Blanks are also useful for making your source file easy to read. For example, use blanks before a subkeyword to indent it and distinguish it from the main keyword.
- Alphabetic characters are case insensitive; i.e., you can use either upper or lower case.

Semantic Rules

Semantic rules test the contents of a parameter for logical values. The basis for semantic rules include:

- checks between parameters within a given object, including conditional checks (e.g., the low limit value must be less than the high limit value)
- cross checks between objects (e.g., software object must be on the same NCM as its hardware object)
- checks against what is already in the archive database (e.g., before defining the system name, an NCM referenced by the system name must exist)

Violation of a semantic rule will generate either an Error or a Warning. The applicable semantic rules are listed for each keyword in the *Reference* section.

Refer to the discussion of *Error Handling* in the *Compiler* section for additional explanation of semantic rules.

Example

The following example shows four source files (Workstation, Global, Model, and NC), and highlights the major parts, such as the DOS file name, file header, comments, keywords, and parameters. Notice that only one "@" is allowed per source file.

Following the source file example is a block diagram of the hypothetical building.

File---> Keyword

Main ----> Keyword

Comments -->

Main ----> Keyword

Subkeyword-->

```
@NET
*  XYZ BUILDING
*  1670 BROAD ST
*  ANYWHERE USA
*
*    Port name (parameter type-name)
*      |  Port used (parameter type-integer)
*      |    | Baud rate (optional,
*      |    |  |          unspecified)
PORT "LPT1",3
*
*     Network name (parameter type-name)
*        |   Description (parameter type-char)
*        |      | This workstation's name
*        |      |               |
NET "XYZ-BLDG","XYZ-BUILDING", "PC1"
   *
   *      PC-Subnet address
    *          | PC-Node address
    *          | |
     N1DIRECT  1,101
*
```

Figure 1: Workstation File for XYZ Building

File--> Keyword

```
@GLOBAL "XYZ-BLDG"
*
*  DEVICES
*
*  NCM name
*   |   NCM description
*   |       |       Graphic symbol #
*   |       |       | Operator instr #
*   |       |       | | Subnet addr
*   |       |       | | | Node addr
*   |       |       | | | | Port 1 type
*   |       |       | | | |  | Port 2 type
*   |       |       | | | |  |    | NCM text
language
*   |       |       | | | |  |    |    |   NT baud
*   |       |       | | | |  |    |    |   |
S2/JC85/
*   |       |       | | | |  |    |    |   | ABDH
baud
*   |       |       | | | |  |    |    |   |   |
NC "NC1","NC1  4th",0,0,1,1,,
NC "NC2","NC2 20th",0,0,1,2," ","JC85","eng",9600,9600
NC "NC3","NC3 35th",0,0,1,3,"S2",,"fre",4800,19200
NC "NC4","NC4 36th",0,0,1,4,"N2","L2"
*
* Workstation name
*    | Workstation description
*    |      |    Graphic symbol number
*    |      |       |Operator instruction
*    |      |       | |
PC "PC1","Lobby PC",0,0
    *
    * Subnet Address
    *        | Node address
    *        | |
    N1DIRECT 1,101
*
PC "PC2","Maintenance Office",0,0
    N1DIRECT 1,102
*
*    Printer name
*     |   Printer description
*     |       |    Graphic symbol number
*     |       |             | Oper instr
*     |       |             | |
PTR "LPTR1","Lobby Printer",0,0
    *
    * Workstation name
    *          |   Port
    *          |      |  Baud rate
    *          |      |   |
    *          |      |   |Driver name
    *          |      |   ||Printer type
    *          |      |   |||
    PCDIRECT "PC1","LPT1",,,
*
PTR "LPTR2","Maintenance Printer",0,0
       PCDIRECT "PC2","LPT1",,,
```

Main------> Keywords--> Trailing commas are optional.

Main----> Keyword

Subkeyword----->

Trailing commas are optional.

```
*
* REPORT DESTINATIONS
*
RPT 1,"HARDWARE"
       DEST "CRITICAL",1,"PC1"
       DEST "CRITICAL",1,"PC2"
       DEST "CRITICAL",1,"LPTR1"
       DEST "CRITICAL",1,"LPTR2"
       DEST "STATUS",1,"LPTR1"
       DEST "STATUS",1,"LPTR2"
       DEST "FOLLOWUP",1,"PC1"
*
RPT 2,"FLOORS"
       DEST "CRITICAL",1,"PC1"
       DEST "CRITICAL",1,"PC2"
       DEST "CRITICAL",1,"LPTR1"
       DEST "CRITICAL",1,"LPTR2"
       DEST "STATUS",1,"LPTR1"
       DEST "STATUS",1,"LPTR2"
       DEST "FOLLOWUP",1,"PC1"
*
RPT 3,"AHUS"
    DEST "CRITICAL",1,"PC1"
    DEST "CRITICAL",1,"PC2"
    DEST "CRITICAL",1,"LPTR1"
    DEST "CRITICAL",1,"LPTR2"
    DEST "STATUS",1,"LPTR1"
    DEST "STATUS",1,"LPTR2"
    DEST "FOLLOWUP",1,"PC1"
*
*  GROUP NAMES
*
*      Group name
*       |    Group description
*       |            |
*       |            |
GRP  "FLOOR-1","FIRST FLOOR",
GRP  "FLOOR-2","SECOND FLOOR",
GRP  "EAST","EAST AREA","FLOOR-1"
GRP  "WEST","WEST AREA","FLOOR-1"
*
*  SYSTEM NAMES
*
*   System name
*       |   System description
*       |        |     NCM name
*       |        |       | Access report group
*       |        |       |  |   Group
*       |        |       |  |    |
SYS "LOBBY","Lobby","NC1",2,"FLOOR-1\EAST"
SYS "GND-FLR","Ground Floor","NC1",2,"FLOOR-1\WEST"
SYS "AC3","AC-3 System","NC1",3,"FLOOR-2"
SYS "AC4","AC-4 System","NC1",3,"FLOOR-2"
SYS "NC1-HW","NC1 Devices","NC-1",1,
*
*
```

Figure 2: Global File for XYZ Building

File---->
Keyword

Comments---->

Main Keyword---->

Subkeyword--->

```
@MODEL
*-----------------
* Software Models
*-----------------
*
*
*      Model name
*          |  Hardware type
*          |       |
CSMODEL "AHU-32","AHU"
*
*             AI group title
*                |
   AITITLE "Analog Inputs"
   BITITLE "Binary Inputs"
   BDTITLE "Binary Data"
*
*    Hardware reference
*        |  AI can be overridden?
*        |    | AI can be adjusted?
*        |    | |   AI descriptor
*        |    | |      |    AI units
*        |    | |      |       |
   CSAI "AI1",N,N,"ZN TMP 5","DEGF"
   CSAI "AI2",N,N,"ZN TMP 6","DEGF"
   CSAI "AI3",N,N,"AT TMP 3","DEGF"
   CSAI "AI4",N,N,"AT TMP 4","DEGF"
   CSAI "AI5",N,N,"DIS HUM","PCT"
*
   CSBI "BI1",N,N,"LO STAT","NOR","LOW"
   CSBI "BI2",N,N,"HI STAT","NOR","HIGH"
   CSBI "BI3",N,Y,"FIRE ALM","NOR","ALARM"
*
   CSBD "BD9",N,N,"OCCUPIED","UNOCC","OCC"
*
*
```

Figure 3: Model File for XYZ Building

File--> Keyword

```
@NC "XYZ-BLDG","NC-1"
*--------------------------
* Hardware Devices per NCM
*--------------------------
* NC-1 HARDWARE
*
*    System name
*     |    Object name
*     |       |    Expanded ID
*     |       |       |
XM "NC1-HW","XBN1","XBN-in-EN-1"

    *
    *     N2 Device address
    *     | N2 trunk number (opt. 1 or 2)
    *     | | XM type(1=XBN,2=XRM,3=XRL)
    *     | | |Poll priority
    *     | | | |
  ADDRESS 2,1,1,1
    *
    *   Symbol number
    *       | Operator instruction
    *       | |
  GRAPHICS 0,0
    *
    *  Auto dial out
    *    | Comm disable
    *    | |
  REPORT N,N
*
XM "NC1-HW","XRL4","Left XRL in EN-2"
    ADDRESS 4,1,3,1
*
*   System name
*      |    Object name
*      |       |    Expanded ID
*      |       |         |
DCM "NC1-HW","DCM-1","DCM in EN-1"
    *
    *   N2 Device address
    *     | N2 trunk number(opt. 1 or 2)
    *     | | Poll priority
    *     | | |
  ADDRESS 1,1,1
    *   Symbol number
    *       | Operator instruction
    *       | |
  GRAPHICS 0,0
    *
    *  Auto dial out
    *    | Comm disable
    *    | |
  REPORT N,N
```

```
DCM "NC1-HW","DCM6","Right DCM in EN-2"
   ADDRESS 6,1,1
*
*----------------------
*Binary Input Objects
*----------------------
BI "AC3","SF-S","SUPPLY FAN STATUS"
   HARDWARE "NC1-HW","DCM-1"
   GRAPHICS 0,0
   DCMHW 3,1,1,2,y
   UNITS "OFF","ON"
   INIT N,0,30
   REPORT n,y,n,n,,5,2,3
*
*----------------------
*Binary Output Objects
*----------------------
BO "AC3","SF-C","Supply Fan Start/Stop"
   HARDWARE "NC1-HW","DCM-1"
   GRAPHICS 0,0
   DCMHW 1,3,200,y
   UNITS "OFF","ON",n
   TIMER 5,1,0,255
   RESET y
   REPORT n,y,n,n,0,5,2,3
   FEEDBACK "AC3","SF-S",y
*
*----------------------
* Analog Input Objects
*----------------------
AI "AC3","RA-T","RETURN AIR TEMP"
   HARDWARE "NC1-HW","DCM-1"
   GRAPHICS 0,0
   DCMHW 1,1
   UNITS "DEG F",0
   LINEPARM -49.929,89.144,-4787.6,221.94
   REPORT n,y,n,n,0,0,5,3,2,4
*
*----------------------
* Analog Output Objects
*----------------------
AOD "AC3","SF-VDF","SUPPLY FAN VFD"
   HARDWARE "NC1-HW","DCM-1"
   GRAPHICS 0,0
   DCMHW 1
   UNITS "% SPD",1
   INIT y
   SPANS 0,100.0,100.0,300.0
   REPORT n,y,n,n,0
```

Figure 4: NC File for XYZ Building

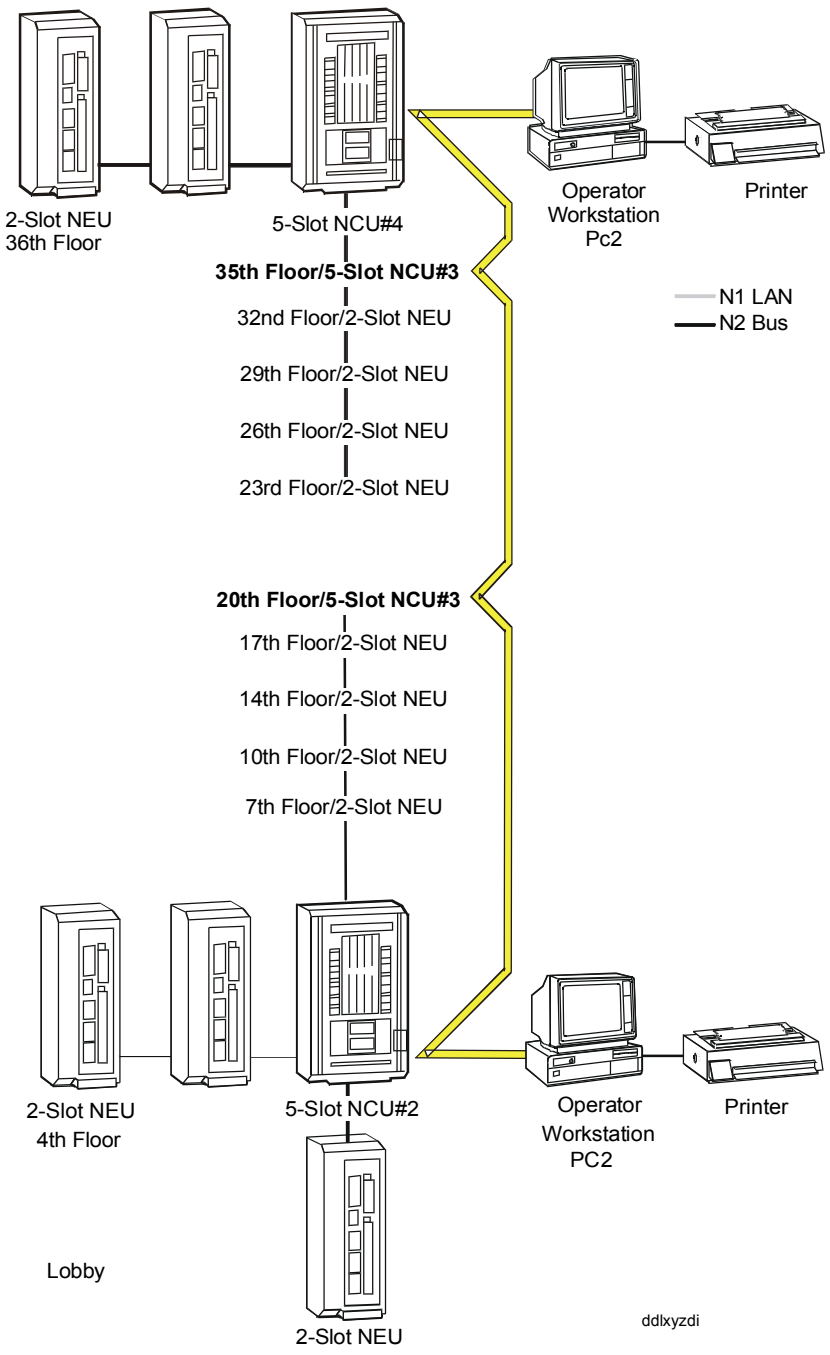


Figure 5: XYZ Building System Diagram

DDL Programmer's Manual

Compiler

This section begins with a discussion of several user aspects of the DDL compiler: the source of its input, how to add or delete objects, the difference between full and incremental compiles, error handling, and referencing.

These sections are followed by *BEFORE Executing the Compiler* and *Executing the Compiler*, step-by-step procedures for setting up and running the compiler from either a branch workstation or job site.

The last part of this section, *Operator Workstation Disk Layout for DDL*, is optional reading, providing operators and programmers with a reference concerning details about DDL's file structure.

Overview

Input to the DDL compiler comes either from an ASCII text editor or via the CAE tool. The syntax for both is identical; however, only the CAE tool generates information used to produce the GPL source file.

To ensure that required directories are created before running a compilation, and since each database builds on the previous database, compile the Workstation Network/Port, Global, and NC files in that specific order. The Model File must be compiled before the NC File that contains CS objects referencing models. However, the Model File can be compiled before or after the Workstation and Global Files.

The Model File and software model database are designed such that on a branch workstation with multiple jobs, only one Model File need be compiled for all jobs on the workstation. You also have the option of compiling one Model File for each job.

Note: In order to use the DDL compiler, you must have basic personal computer skills and an understanding of DOS concepts, such as how to create files and execute programs.

How the DDL Compiler Works

The DDL compiler operates in a batch mode (source files processed one at a time) using the following sequence:

1. Reads the input file, one line at a time and performs syntax checks (e.g., is the parameter the correct type). If no errors are discovered, that section is added to the record.
2. Accumulates information in a record for the key and all its subkeys.
3. Checks semantic rules when the key and all its subkeys have been through Steps 1 and 2.
4. Adds the record to the database.
    - If the compiler does not detect any errors, it generates and updates the archive database.
    - If it detects errors, the compiler generates error messages in a list file and leaves the database unchanged.
5. Returns a DOS exit code at the end of the compile. The exit code is 0 if no errors are found during the compile, and 1 if errors are found. This exit code is useful if you are using DOS batch streams to run the compiler.

There is no interaction with the user while a file is compiling. For a description of the screen during a compilation, refer to *Executing the Compiler*.

Demand Limiting/Load Rolling Databases

Since Release 6.0, DDL supports adding (compiling) DLLR (Demand Limiting/Load Rolling) databases, and defining DLLR loads. Load definition is part of the object's definition. (You define an object as a load when you define the object.) The objects that can be Demand Limiting/Load Rolling loads are BD, BO, MSD, MSO, and MC.

What the DDL Compiler Does Not Do

Feature Databases

The DDL compiler does not generate most feature databases (except DLLR, TIMEZONE and CARD Access features). Feature databases include Trend, Totalization, Scheduling, and Password.

DDL also does not generate databases used by features such as Graphics, Operating instructions, and Summary/Report Scheduling.

JC-BASIC Processes

DDL does not create JC-BASIC processes. However, it is possible to add process *names* to the database by using the JCB keyword.

The JCB keyword creates the system\object name of a process. This name exists in the OBJENT database (object entry database) only. There is NO process object code associated with the name when the name is created by a DDL compile.

As with other objects, during a full compile, DDL deletes all process objects. However, DDL only deletes all database references to process objects; it does not delete the process object file, which resides in the directory:

FMSDATA\[NETWORK]\DEVICES\[NCn]\PROCESS\[SYSn].

Deleting Obsolete Directories

During a recompile, the compiler does not delete obsolete directories or obsolete files in these directories. In other words, if the recompiled database does not use the same directory structure as the original, the old, obsolete directories will not be automatically deleted. You must manually delete the unnecessary directories and files, using the DOS RD (Remove Directory) and DEL (Delete) commands.

Note: DDL does not automatically delete these directories and files because they may contain drawings or other files (not created by DDL) that you may want to keep.

Adding and Deleting Objects

To add new data to an existing Workstation, Global, Model, or NC database, perform an incremental compile.

To delete data from an existing Workstation, Global, or Model database, it is necessary to edit the original source file (deleting the unwanted data) and perform a full compile of the edited file and all files dependent on the file. For example, if an @NET File is recompiled, all @GLOBAL and @NC Files must be recompiled. If an @MODEL File is recompiled, all @NC Files with CS objects must be recompiled.

Alternately, online generation (PMI) can be used to delete information and update the archive database with an upload.

To delete objects from an existing NC database, use the DELETE keyword during an incremental compile (i.e., the incremental compile can add to any database, but can only delete from the NC database).

Notes about Deleting Objects

- When a software object is deleted, the compiler does not check to make sure that it is not referenced by another object or feature.
- When deleting software objects occupying a slot (or slots) in a hardware device, those slots will be marked “free” in the archive database.
- If any of its slots are currently in use, you may not delete a hardware object.
- If the object being deleted does not exist, the compiler generates an error.

To delete Access features, use DELCARD and DELTZ keywords during an incremental compile.

Incremental vs. Full Compile

You can invoke the compiler as either an incremental or full compile.

Full Compile

A *full compile* may be performed on any file type. Its purpose is to:

- create the original archive or OWS-specific database
- delete and fully replace any existing database that the source file generates. This means that any NC or Global uploaded changes to the archive database (except feature data not generated by DDL) will be lost unless the same changes are reconstructed in the source files. Further, any OWS-specific databases will contain only the information supplied in the DDL source file. For this reason, full compiles should be invoked with care.

Since the DDL compiler does not generate most feature databases, a full DDL compile does not disturb those feature databases uploaded to the archive disk that DDL cannot generate. However, after a new compile, the undisturbed feature databases may not coordinate with the object database. A typical example of this is an existing schedule for a currently non-existent object.

How a *Full* Compile Affects Demand Limiting/Load Rolling Databases

Release 6.0 made some noticeable changes to Demand Limiting/Load Rolling. The Demand Limiting/Load Rolling Group is treated by DDL like any other object; that is, DDL can generate a DLLR database on a full compile, and an incremental NC compile can use the DELETE keyword to delete a specific DLLR object. The DDL compiler also provides a means to define DLLR loads. A DLLR load can be defined using the subkeys available to BD, BO, MSD, MSO, and MC objects.

Incremental Compile

An *incremental compile* may add information to any database and delete information from NC databases. The items that were in the database before the compile will be left intact, allowing additions to the archive database without overwriting any previous changes uploaded from the online database.

Indicate an incremental compile by adding a "+" to the file keyword at the beginning of the source file (i.e., @NET +, @GLOBAL +, @MODEL +, or @NC +). When creating the source file for an incremental compile, only include the keywords and parameters needed to make the additions or deletions desired.

Incremental Compile Considerations

- The same error checks made by a full compile are made by an incremental compile. If an error occurs during a compile, the existing database is unchanged.
- Deletion (using the DELETE, DELCARD, and DELTZ keywords) is only allowed in incremental compiles of NC source files.
- After an incremental compile where objects have been deleted, the undisturbed feature databases may not coordinate with the object database. A typical example is an existing schedule for a deleted object.
- After an incremental compile where objects have been deleted, there may be software objects that still reference the deleted object. A typical example is an existing BO may reference a deleted BI object as feedback.

Error Handling

The compiler can generate error messages and warnings.

Errors and Warnings

Error messages indicate problems with parameters that prevent the compiler from building the object requested. If any error messages occur in the source file, none of the objects are built and the archive database is left as it was before the compile started. You must correct the errors and recompile the file.

Warnings indicate that there is information missing from the database, or that guidelines for optimal operation have been violated. A compile with warnings is successful (updates the database) as long as the compile does not have errors. Most NC warnings are due to referencing software objects that do not exist in the database.

A full compile and an incremental compile make the same error checks.

Locating Errors and Warnings

Errors and warnings are listed in the list file. The list file, which also includes the lines of source generated during the compile, is explained under *Compiler List File*.

All syntax errors will be listed immediately following the line with the error (in the list file). Semantic errors will be listed at the end of the entire keyword (the end of a keyword is determined when the compiler reads a line with a main keyword or reaches the end-of-file). Second pass errors are located near the end of the list file.

Note: *Multiple Errors*. The DDL compiler is guaranteed to find the first error on a line, and attempts to issue all errors for the line. However, detection of a first error may cause DDL to miss other errors or flag extra errors. For example, if a closing quote is missing on a string parameter, the DDL compiler may not recognize any more parameters on the line because it reads all characters until it finds another double quote or carriage return.

Note: *ERRORLOG.TXT File Errors*. In addition to errors appearing in the list file, some errors result in messages in the ERRORLOG.TXT file. Examples of these errors are:

```
^Database error
^Error opening source file, cannot
continue
^DBM Init Failed, cannot continue with
DDL compile
```

For information on the location of the ERRORLOG.TXT file (determined by the ERRORLOGPATH variable of the WIN.INI file), see the *WIN.INI File* heading later in this section.

The DDL compiler is a two-pass compiler. The first pass reads the source file line by line. Upon reaching the end of the file, the DDL compiler begins its second pass. The second pass of the compiler checks for database relationships.

The second pass of an @NC compile searches for the existence of system\objects referenced during pass one (e.g., feedback for BO) that did not exist at the time of reference. If a software object referenced by another software object does not exist, DDL generates a warning near the end of the list file. If the reference now exists (entered later in source file), and a semantic rule was violated (e.g., BI referenced, but object is an AI), DDL generates an error near the end of the list file.

The second pass of a @GLOBAL file checks that the OWS (if any) named in the network definition is defined in the global database and has the same address, if N1 Direct.

Software-to-Software References

Software references are those references made by one software object to another. One example would be a Binary Output (BO) object that specifies the name of its Binary Input (BI) feedback object.

The DDL compiler checks for the referenced object's existence and correct type. A warning occurs at the end of the list file if the object is not in the database. If the referenced object is there but is the wrong type, the compiler generates an error message.

Here are examples of software-to-software references.

- BO referencing a BD or BI feedback object
- AOS referencing an AI, ACM, or AD feedback object
- an AD, BD, MSD, or MC referencing an associated object
- a BD, BO, MSD, MSO, or MC referencing a DLLR object
- a BD, BO, MSD, MSO, or MC referencing a BD, BI, MSD, or MSI as a load comfort object
- a DLLR object referencing an MSD or MSI target object, an ACM, AD, or AI meter object, or a BI EOI object
- PID Loop input referencing an AI on the same DCM, or a PID Loop output referencing an AOD or PID Loop on the same DCM. (For these PID Loop examples, an error message occurs if the referenced object is not on the same DCM.)
- an MC referencing a SUBORDINATE object

Software-to-Hardware References

Software-to-hardware references occur when you define a software object and list the L2, N2, S2, or LON device (by system\object name) on which it resides. An error occurs if the hardware object does not exist, if it is the wrong type, or if it is located on a different NCM. Therefore, you must generate the hardware object before the software object, either in a previous compile (incremental only) or earlier in the current compile.

As part of this check, the compiler verifies that any slot number required by the new software object is unused.

CS Object-to-Software Model References

CS object-to-software model references occur when you define a generic Control System object in the NC File. The CS object definition must reference an existing software model. The DDL compiler checks the model database for the referenced software model when it compiles the NC File. An error occurs if the specified model does not exist.

Feature-to-Object References

Feature-to-object references occur when you define a feature and specify the object the feature is associated with. An error occurs if the object does not exist. Therefore, you must generate the object before the feature, either in a previous compile (incremental only) or earlier in the current compile. Examples: CARD reference to a D600. (The D600 must be compiled first.) TIMEZONE reference to a D600. (The D600 must be compiled first.)

Software-to-Feature References

Software-to-feature references occur when you define a software object and reference a feature the software object uses. An error occurs if the referenced feature does not exist. Therefore, you must generate a referenced feature before the software object that references it, either in a previous compile (incremental only) or earlier in the current compile. Examples: READER reference to a TIMEZONE. (The TIMEZONE must be compiled first.) BI (on D600) reference to a TIMEZONE. (The TIMEZONE must be compiled first.)

Duplicate Objects

You cannot use DDL to add an object that is already in the database (i.e., there is no "modify" or "replace" capability). You must first delete the object. This is true for all databases.

Duplicate Addresses

The DDL compiler does check to ensure that a given NC port is not used by more than one printer or Operator Workstation.

The compiler *does* check the following:

- N2/S2/L2 addresses of all devices on the same trunk must be unique (Also applies to LONWORKS compatible devices if address is not 0.). For example, an XRL cannot be assigned a trunk address already occupied by a DCM.

- More than one software object (e.g., BI, BO) cannot be assigned to the same S2, N2, and L2 hardware device slots. An error message will be issued when necessary. This means if more than one object is assigned to the same slot of a particular XRL (for example), the compiler will flag the error.

Conditionals

Conditional fields are handled as follows:

- If an Optional conditional is entered when not needed, the compiler ignores the parameter value. No error message or warning results.
- If an Optional conditional is not entered when needed, the compiler uses the default value. Normal operation continues.
- If a required conditional is entered when not needed, the compiler ignores the parameter value. No error message or warning results.
- If a required conditional is not entered when needed, the compiler generates an error message.

References Within Global Files

Global references occur when you define a global item and associate it with another global item. An error occurs if the associated global item does not exist. Therefore, you must order your Global source file such that global items are defined before they are referenced.

Global definitions must occur in the following order:

1. An NCM must be defined before being referenced by:
   a. systems
   b. Operator Workstations connected via NC-dial or NC-direct
   c. printers connected to that NCM
2. Printers or Operator Workstations must be defined before being referenced as a destination of an Access Report Group.
3. PC Files (Access Report Group destination files) must be defined before being referenced by Default Destination as original or default files.
4. PC Groups must be defined before they can be referenced by systems or lower level groups.

5. Access Report Groups must be defined before being referenced by systems.

BEFORE Executing the Compiler

The Release 8.0 or later compiler and decompiler will only work on Release 6.0, 7.0, or 8.0 databases. If the compiler or decompiler is invoked and the database is at an earlier release, the compile or decompile will not occur and you will be instructed to convert the database. See the *Operator Workstation Technical Bulletin i*n the *Metasys Network Technical Manual (FAN 636)* for instructions. After the conversion is complete, you are free to compile or decompile the converted database.

The following procedures are not required when compiling on an Operator Workstation at the job site where the workstation software is already properly installed. They *are* required at the branch workstation, when there will be more than one archive database (FMSData) supporting multiple job sites.

Before executing the compiler, you must do the following at the DOS prompt (without Windows running in the background):

- Set the FMSDOS environment variable for the job you are compiling. Make sure the directory that the FMSDOS environment variable points to exists, and make sure it contains the WIN.INI file for the job you are compiling. Create the directory, if necessary.

  There is a utility called PREP-FOR.BAT that automates the setting of environment variables. See the *Move Utility Technical Bulletin* in the *Metasys Network Technical Manual (FAN 636)* for more information.

- Set the WIN.INI file variables.

- Make sure the directories that the WIN.INI file variables point to exist.

These procedures are described below. Examples for both job site workstations and branch workstations are given.

Setting the FMSDOS Environment Variable

The path name you specify when setting this variable must point to the DOS directory that contains the WIN.INI file for the job you are compiling.

If there is more than one job, each job will have its own directory containing its own WIN.INI file. When you are getting ready to compile DDL files for a job, you must set the FMSDOS environment variable for that job.

To set the FMSDOS environment variable, type the following at the DOS prompt:

SET FMSDOS = [pathname containing WIN.INI file for job]

Job Example:

```
SET FMSDOS = C:\JOBXYZ
```

Branch Example:

```
SET FMSDOS = C:\PROJECT\90120001
```

Make sure the named directories exist, and make sure they contain the WIN.INI file for the job. Create the directories, if necessary, using the DOS MD (Make Directory) command.

Job Example:

```
MD C:\JOBXYZ
```

Branch Example:

```
MD C:\PROJECT\90120001
```

(Or use an existing project directory created by the contract drawing manager.)

Setting WIN.INI File Variables

Use an ASCII editor to set the following variables in the WIN.INI file. Each job will have its own WIN.INI file, typically in its FMSData directory. The FMSData directory also contains the archive database for the job.

- Data

This variable specifies the directory where the Metasys FMS (Facility Management System) software is installed.

Job Example: **`Data = C:\FMS`**

Branch Example: **`Data = D:\METASYS`**

- FMSData

This variable (maximum of 20 characters) specifies the directory where the archive database for the job is located. Each job will have its own directory for its own archive database.

Operator Workstations at the job site may choose to place this variable in the AUTOEXEC.BAT file. (To set this variable at installation through the configuration file, see the *Operator Workstation Technical Bulletin* in the *Metasys Network Technical Manual, Volume 1*.)

Job Example: `FMSData = C:\FMS\DATA`

Branch workstations will have to set this variable for each job.

Branch Example:

FMSData = C:\PROJECT\90120001

- Models

This variable specifies the directory where the software model databases reside. Use of this variable will determine whether the software model database is shared between jobs on a branch workstation, or whether each job has a unique software model database. When the same directory is referenced in each job's WIN.INI file, the software model database is shared.

Job Example:

```
MODELS = C:\FMS\DATA\MODELS
```

Branch Example 1:

```
MODELS = C:\PROJECT\90120001\MODELS
```

(This example shows a unique software model database for the 90120001 job.)

Branch Example 2:

```
MODELS = C:\PROJECT\MODELS
```

(This example shows how several jobs can share a software model database. All jobs sharing the software model database would have the identical MODELS = line in their WIN.INI files.)

- ErrorLogPath

This variable specifies the directory containing the file ERRORLOG.TXT. The FMS software will place internal error messages encountered during program execution into this file.

Job Example: `ErrorLogPath = C:\FMS\ERROR`

Branch Example:

```
ErrorLogPath = C:\PROJECT\90120001\ERROR
```

- Language

The language choice is ENGlish.

Example: `Language = ENGlish`

Make Sure All Directories Exist

Make sure all the directories pointed to by the WIN.INI file variables exist. If necessary, create the directories using the DOS MD (Make Directory) command.

In particular, the FMSData directory must exist because the DDL compiler places all of the databases it generates into the FMSData directory, except for the model databases (which are placed in the directory pointed to by the Models variable).

The last directory pointed to by the Models variable need not exist; it will be created when the Model file is compiled. However, the preceding directories must exist. For example:

```
MODELS = C:\PROJECT\JOB100\MODELS
```

In this example, you must make sure the PROJECT and JOB100 directories exist. The Models subdirectory does not have to exist because it will be created when the Model file is compiled.

Note: Shipped with the system are fixed databases that must be present for DDL to compile. Refer to *Operator Workstation Disk Layout for DDL* section in this document for further information.

Executing the Compiler

Execute the DDL compiler only after performing all of the necessary procedures described in *BEFORE Executing the Compiler*.

The DDL compiler runs as a program under the DOS operating system. You specify a source file name to be compiled and the compiled output will be added to the archive database.

Be sure to follow the proper order for compiling the various file types since each one creates directories used by the next level down. The proper order is: Workstation, Global, Model, NC.

This section lists the necessary steps to run the DDL compiler. The *Operator Workstation Disk Layout for DDL* section, which follows, discusses the nature and location of all files referenced in the following steps.

1. Bring the Operator Workstation up in DOS without Windows running in the background.
2. At the DOS prompt, type the command that invokes the compiler:

   ```
   DDL sourcefilepath [.DDL]
   ```

   where:

   - DDL is the command that invokes the DDL compiler.
   - Sourcefilepath [.DDL], is the file path name of the DDL source file to compile. If the file name has no leading backslash (\) or drive designation (C:), the file path starts in the current working directory.

   Example: `DDL ncone.ddl`

   The source file path may point to any location on any accessible disk for the DDL source file. *Operator Workstation Disk Layout for DDL* shows a suggested source file structure to help maintain consistency.

Example: `C:\FMS\DATA\NET.DDL`

You may omit the file extension because the .DDL extension is assumed. If you include the source file extension, then it must be .DDL. Any other extension causes an error message.

The entire command line is case insensitive (upper or lower case allowed).

Example: `C:\FMS\DATA\NET.ddl`

Database Output

The DDL compiler outputs to the Operator Workstation disk those data records that make up the archive database and the OWS-specific database. The FMSData variable, specified in the WIN.INI file, indicates the starting path for the database subtree (except for the software model database, which is specified by the Models variable).

The network name from the @GLOBAL keyword indicates the network under which the global databases go when compiling the Global File.

The network name and NC name from the @NC keyword indicate the network and NC subdirectories to which the NC databases are to go.

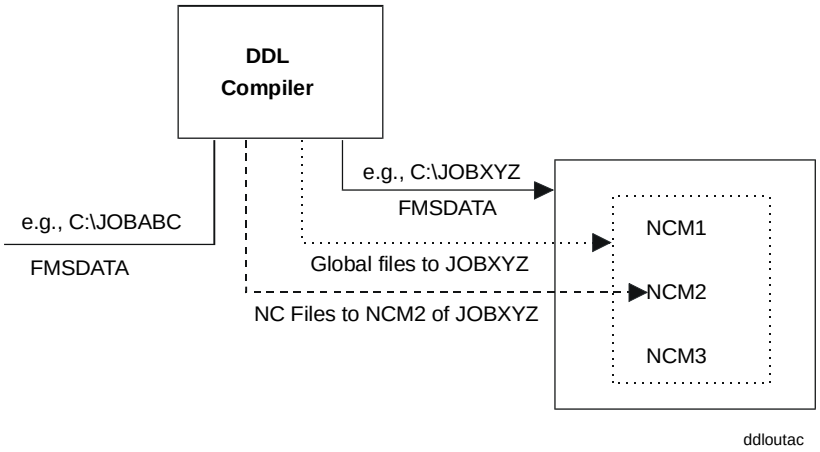


Figure 1: Output to Archive Database

Screen Interaction

Although there is no interaction with the compiler once it is invoked, you can monitor the compilation by observing the screen. When a compilation begins, the necessary copyright statements appear, followed by the word “Working.” There may be a brief pause while the database is saved (so it can be restored if any errors occur) after “Working” appears. Once the database is saved, “Working” will be followed by a series of dots (periods). One dot lists for each main keyword compiled, and one dot lists for each second pass reference checked.

```
Working.....................................
.................
```

Main keywords with an error in them list as “*” instead of a dot; warnings list as “?”.

```
Working......*..*.........................
..........*............*.....?..
```

When the compiler finishes, it indicates the compile status on the screen of the Workstation that invoked it, and shows the number of errors and warnings (“None” displays if the compile was successful). Therefore, you do not need to examine the list file to determine if the compile was successful.

```
Working.....................................
DDL compile complete
Errors: None
Warnings: None
No Errors found compiling DDL sourcefile, database
created/modified
```

The completion status also indicates whether the archive database was created (or updated).

You can call a usage statement by entering “ddl ?” or just “ddl” <CR>.

```
C:\>ddl ?
Usage: ddl sourcefilepath[.ddl]
```

Compiler List File

The DDL compiler generates a list file as part of the compile process. The list file contains the DDL source lines and any errors or warnings that the compiler found. Following the necessary copyright information, the list file appears as shown below:

```
                    @NET
                    *  XYZ BUILDING
                    *  1670 BROAD ST
                    *  ANYWHERE USA
                    *
                    *  Network name (parameter of type-name)
                    *       | Description (parameter of type-char)
                    *       |          | This workstation's name
                    *       |          |                       |
                    NET "XYZ-BUILD","XYZ-BUILD-NETWORK","PC1"
Error Message-->    ^ Parameter 1 invalid name
                       *
                       *  PC-Subnet addr.(parameter type-integer)
                       *          | PC-Node address
                       *          |  |
                       N1DIRECT  1,101
                    *
                    *   Port name (parameter of type-name)
                    *       |  Port used (parameter of type-integer)
                    *       |     |Baud rate (opt parameter,
                    unspecified)
                    *       |     | |
                    PORT "LPT1",6
Error Message-->    ^ Parameter 2 value out of range
                    *
Summary-->          Pass1 failed
                    2 error(s) found
                    0 warning(s) found
```

The list file is created with the same file path as the source file, but has a .DLT extension. Errors and warnings are summarized at the end of the list. The example above showed two errors and no warnings. A summary for a warning follows.

```
Pass1 completed successfully!
WARNING AC3\SF2 does not exist, referenced by AC3\SFC
Pass2 completed successfully!
0 error(s) found
1 warning(s) found
```

The error messages appear in the language chosen by the Language variable of the WIN.INI file. The default is English.

Downloading and Uploading

A download or upload transfers data from one device or file to another. This transfer of data is necessary so that information at different locations is synchronized and kept current.

The archive database should always be a current copy of the operational database. For example, if you make changes to the archive database by executing an incremental DDL compile, you should copy the archive database to the operational database by executing a download. Conversely, if you make changes to the operational database, you should copy the operational database to the archive database by executing an upload.

The four types of downloads and uploads are:

- NCM download
- NCM upload
- global download
- global upload

Note: These downloads and uploads are conducted at the Operator Workstation, under the Network Map Action Menu. For more complete information, refer to the *Operator Workstation User's Manual*.

An *NCM download* copies the archive NC database to the specified NCM. When you install an NCM or change the archive database, you should perform an NCM download.

An *NCM upload* copies the operational NC database to the archive NC database. Whenever you change the operational database, it is important to perform an upload to keep the archive database current.

Note: If an NCM is uploaded to a workstation that does not contain the software model referenced by a CS object in the NCM, the CS object cannot be downloaded. If the NCM is uploaded to a workstation containing a different model definition for the model referenced by the CS object, the CS object will be downloaded with the changes.

A *global download* copies the archive global database to the operational global database at the workstation performing the download. The Operator Workstation automatically synchronizes all global databases via global download to all devices on the network.

A *global upload* copies the operational global database to the archive global database at the workstation performing the upload. Whenever you change the operational database, it is important to perform an upload to keep the archive database current.

Operator Workstation Disk Layout for DDL

This section describes the layout on the Operator Workstation disk for the files relevant to DDL. It is provided as a reference for those programmers and operators seeking details about DDL's underlying file structure. Functional operation of DDL is described in *Executing the Compiler* and in the *Reference* sections.

DDL-related files are located under three major directories. The names and locations of these directories are specified by variables in the WIN.INI file. The three directories are:

- FMS system software directory (DATA variable)
- Models archive database directory (Models variable)
- FMSdata archive database directory (FMSdata variable)

FMS System Software

The FMS system software (Metasys software), including the DDL compiler itself, is installed in a DOS directory. This directory is named by the Data variable in the WIN.INI file. The default name for the Data variable is C:\FMS.

The Data directory contains a number of subdirectories for Metasys system software files (e.g., BIN, STATIC, Language).

BIN Subdirectory

This subdirectory contains the executable files, including the DDL compiler itself. If the default name for the Data variable is used, the DDL executables are in the C:\FMS\BIN directory.

STATIC Subdirectory

The databases that must be present for DDL compilation to occur are in the "Data"\Static directory. They are part of the FMS system software. These databases are fixed and shipped with the FMS system software; you cannot change them.

The default location for the Static subdirectory is C:\FMS\Static.

The static files are listed below.

Note: In the list ??? in a file name indicates the three letter language abbreviation: ENG, FRE, and DEU. For example, “diag???.sbv” indicates there are three files: diageng.sbv, diagfre.sbv, and diagdeu.sbv.

command.dbf	required by GPL and PMI, not DDL
password.dob	required by DDL and PMI
ahu.dbf	required by DDL, GPL, and PMI
attrib.dbf	required by DDL, GPL, and PMI
dc9100.dbf	required by DDL, GPL, and PMI
dr9100.dbf	required by DDL, GPL, and PMI
dx9100.dbf	required by DDL, GPL, and PMI
dx91e.dbf	required by DDL, GPL, and PMI
hdwmodel.dbf	required by DDL, GPL, and PMI
lmtcu.dbf	required by DDL, GPL, and PMI
lmtcua.dbf	required by DDL, GPL, and PMI
loncvt.dbf	required by DDL, GPL, and PMI
lontcu.dbf	required by DDL, GPL, and PMI
lontcua.dbf	required by DDL, GPL, and PMI
mig.dbf	required by DDL, GPL, and PMI
ndm.dbf	required by DDL, GPL, and PMI
n2open.dbf	required by DDL, GPL, and PMI
phx.dbf	required by DDL, GPL, and PMI
tc9100.dbf	required by DDL, GPL, and PMI
untvav.dbf	required by DDL, GPL, and PMI
vma.dbf	required by DDL, GPL, and PMI
vnd.dbf	required by DDL, GPL, and PMI
xt9100.dbf	required by DDL, GPL, and PMI
cardrdr.1	required by PMI
cardrdr.sbd	required by PMI
cardrdr.sbf	required by PMI
diag.1	required by PMI
diag.sbd	required by PMI
diag.sbf	required by PMI
diag???.sbv	required by PMI
optrans.1	required by PMI
optrans.sbd	required by PMI
optrans.sbf	required by PMI
person.1	required by PMI
person.2	required by PMI
person.21	required by PMI
person.29	required by PMI
person.5	required by PMI
person.3	required by PMI
person.sbd	required by PMI

person.sbf	required by PMI
restart.sbp	required by PMI
critical.1	required by PMI
critical.sbd	required by PMI
critical.sbf	required by PMI
maint.1	required by PMI
maint.sbd	required by PMI
maint.sbf	required by PMI
status.1	required by PMI
status.sbd	required by PMI
status.sbf	required by PMI
user???.1	required by PMI
user???.sbd	required by PMI
user???.sbf	required by PMI
head???.sbv	required by PMI
toper???.sbv	required by PMI
tcard???.sbv	required by PMI
rpaut???.sbv	required by PMI
rpaut???.sbp	required by PMI
hcemp???.sbv	required by PMI
menu???.sbv	required by PMI
entry???.sbv	required by PMI
blank???.sbv	required by PMI
hcaut???.sbp	required by PMI
main???.sbp	required by PMI
tstat???.sbv	required by PMI
tmain???.sbv	required by PMI
tcrit???.sbv	required by PMI
hmenu???.sbv	required by PMI
lmenu???.sbv	required by PMI

There are also a number of index files (.ndx and .nob) that are necessary. All of these files are installed when the Operator Workstation installation procedure is followed.

Models Archive Database

The directory containing the Models archive database is obtained from the Models variable in the WIN.INI File. The default name is C:\FMS\Data\Models.

This database can be unique for each job, or it can be shared by multiple jobs. The directory contains the software model data from the DDL compilation of the Model File(s) and from online generation of models at the workstation. For successful download of generic CS objects, the software model referenced by the CS object must exist in the database of this directory.

FMS Archive Databases

The directory containing FMS archive databases is the root of the tree containing all the databases for a single job. Under this directory is the WIN.INI File for the job, and a DDL directory that is the recommended location of the job's DDL Files and the directory where the decompiled files are placed.

This directory is named by the FMSData variable in the WIN.INI File. The default name is C:\FMS\DATA.

FMSDOS Environment Variable

Before running the compiler, you must establish the directory path to the configuration text file (WIN.INI) by setting the FMSDOS Environment Variable. (Refer to the DOS manual for details on environment variables.)

The directory pointed to by the FMSDOS environment variable must already exist and must contain the WIN.INI File for the job you are compiling.

On an Operator Workstation that has the FMS software installed, FMSDOS will already be set up.

On the branch PC, the FMSDOS environment variable must point to the archive database directory for the job you intend to compile. In other words, if you are compiling files for a job whose archive database is located in the C:\project\02099001 directory, make sure the WIN.INI File is also in this directory, and type the following at the DOS prompt to set the FMSDOS environment variable:

```
SET FMSDOS = C:\project\02099001
```

Fixed METASYS\SB4W Directory

The directory C:\METASYS\SB4W is a fixed directory that contains Superbase 4™ definition files. These files are required by the PMI for operator transactions and CARD user data. They are installed on all OWSs. (Since they are not needed by DDL, they are not installed on EWSs.) The files in this directory are:

parms.1	required by PMI
parms.cmd	required by PMI
parms.doc	required by PMI
parms.sbd	required by PMI
parms.sbf	required by PMI

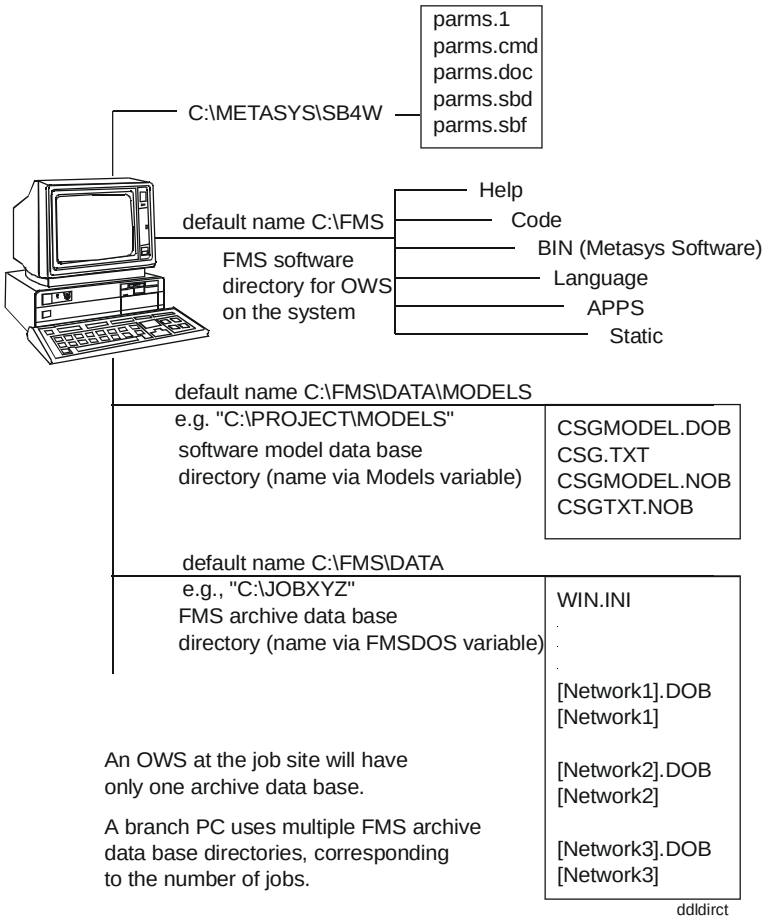


Figure 2: DATA, FMSDATA, and MODELS Directories

WIN.INI File

The WIN.INI file contains several variables used to set up configuration needed by the DDL compiler and other DOS programs belonging to the FMS. One WIN.INI file exists for each FMS (job) you have on your disk.

On the Operator Workstation, if the FMS software has been installed, this WIN.INI file will have been created for you and placed in the directory you specified when you installed the software. The FMSDOS variable will automatically be set to point to the same place.

On the branch PC, you must build your own WIN.INI file for each job and place it in the directory pointed to by FMSDOS; you must also set FMSDOS for each job.

You change the following variables in the WIN.INI file to make the DDL compiler and decompiler function in a specific way.

Data

This variable specifies the directory on which you installed the FMS software.

Example: `Data = C:\FMS`

FMSData

This variable specifies the DOS path name of the directory containing the archive database for the FMS being generated.

Example: `FMSData = C:\JOBXYZ`

Note: The FMSDOS environment variable points to the directory containing the initialization file WIN.INI. The FMSData variable in WIN.INI identifies the directory containing all the archive databases for a job.

Models

Defines the DOS path of the directory containing the file software model database.

Example: `Models = C:\FMS\DATA\MODELS`

ErrorLogPath

Defines the DOS path of the directory containing the file ERRORLOG.TXT. All FMS programs will place internal error messages encountered during program execution into this file.

Example: `ErrorLogPath = C:\FMS\DATA\ERROR`

It is recommended that this DOS path be to the "FMSData" directory or a subdirectory of it. This assures that SAVEDB will back up the errorlog.txt file.

Note: The errors DDL places in the ERRORLOG.TXT File are discussed in the *Compiler* section, under *Locating Errors and Warnings*.

Language

The selection identifies which language files to read to produce error messages and other text messages. Change this if you need the FMS software to generate messages in a different language for each FMS that you can operate in. Only the first three characters of the language are significant. The default value is English.

Example: `Language = ENGlish`

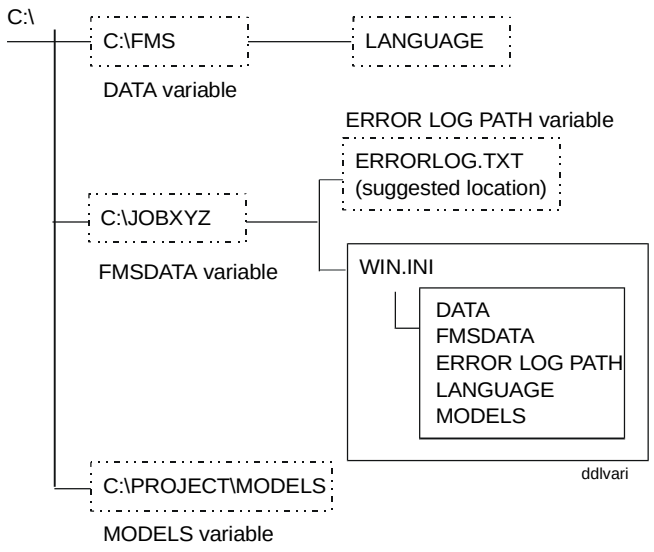


Figure 3: Variables Set by the WIN.INI File

FMSData Files

The FMSdata directory contains the WIN.INI file, various archive and operational FMS databases (the combined output of compiled DDL files and online generation), and the DDL subdirectory, which is the recommended location for the DDL source code files and the directory where decompiled .UND files are placed.

Below is a diagram of the FMSData files. The information inside the brackets (e.g., [NETWORKn]) represents global databases for the specified networks.

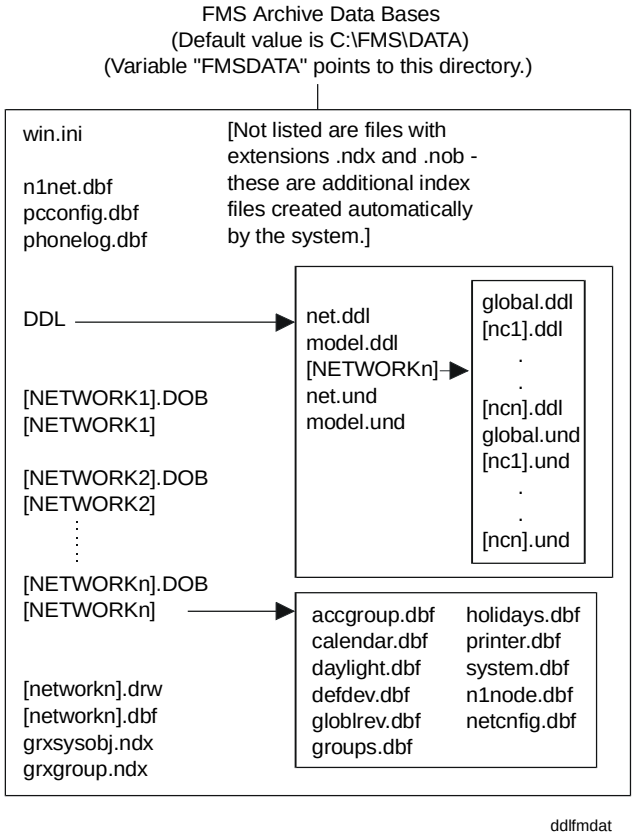


Figure 4: FMSData File Diagram

The hierarchy shown in Figure 4 is a suggested DDL source file layout. It is not mandatory since the DDL compiler accepts any source file name (including path) as the input source file. However, the decompiler will use this directory structure when decompiling a database.

WIN.INI

C:\FMS\DATA is the preferred location for the WIN.INI File since GPL assumes it can be found there. However, since the FMSDOS environment variable locates this field, DDL does not require it to be there.

PCCONFIG and N1NET Files

The DDL compiler outputs these two files when it compiles the NET.DDL File. The PCCONFIG File contains the port definitions for this workstation. The N1NET File contains the list of accessible networks.

DDL Directory

The DDL directory is directly under the FMSdata level. This directory (and its subdirectories) is the recommended location for the DDL source files (.DDL) and the directory where the decompiler will place the decompiled files (.UND) for the FMS.

Directly under the DDL directory are the Workstation source file (NET.DDL) and the Models File (MODELS.DDL). If the decompiler has been used, this is where the decompiled Workstation File (NET.UND) and decompiled Models File (MODELS.UND) are placed.

In addition, directly under the DDL directory is one network subdirectory for each network named in the database. Each network subdirectory is the recommended location for the Global File (GLOBAL.DDL) and NC Files (NCn.DDL) for the network. If the decompiler has been used, this is the location of the decompiled Global and NC Files (GLOBAL.UND and NCn.UND).

Either you or CAE must create the DDL source files, store them in the desired directories, and invoke the DDL compiler.

NETWORKn.DOB

The NETWORKn.DOB directory name is the 8-character network name with a .DOB extension given on the NET keyword line. The files in this directory are the *archive copy* of the global databases for this network, and are generated by compiling the Global File for this network. The included databases are:

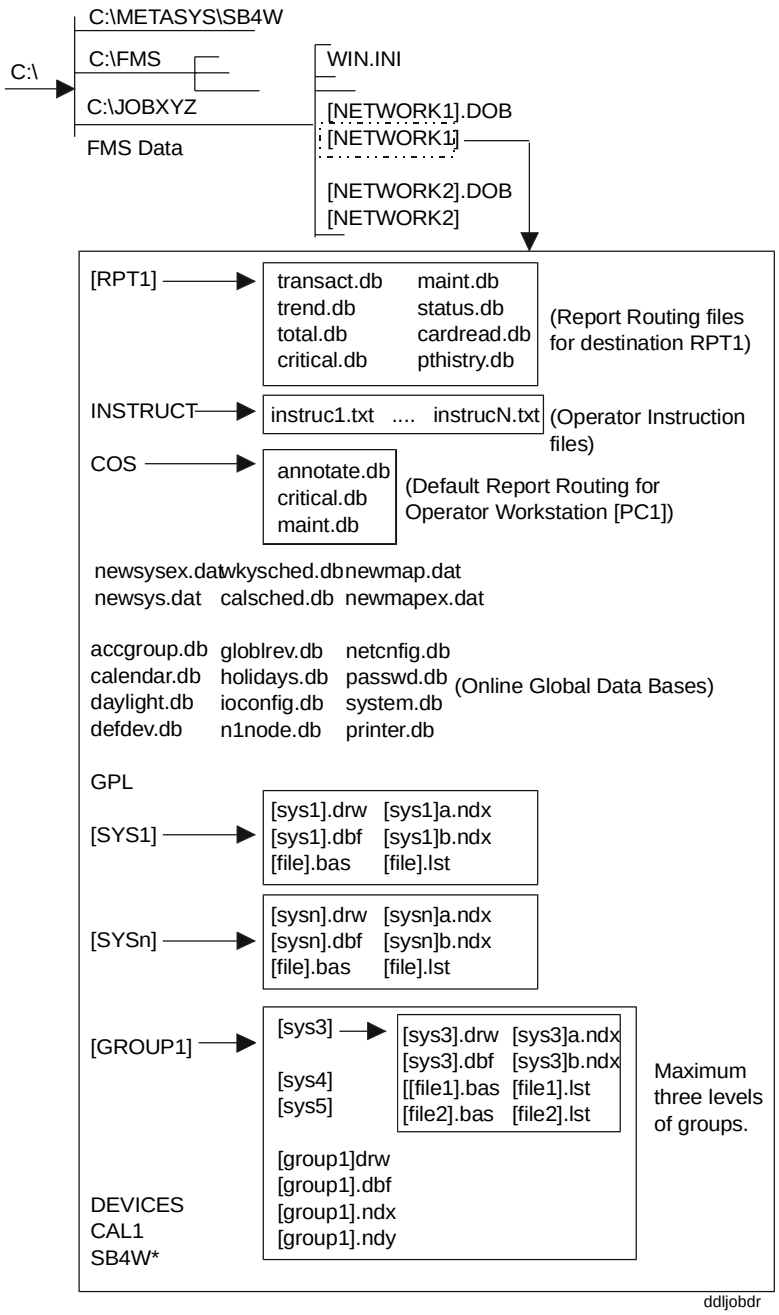
accgroup	access report groups
calendar	alternate periods
daylight	daylight saving time information
defdev	report destination default devices
globlrev	global database revision information
groups	PC groups hierarchy
holidays	holiday periods
netcnfig	N1 directory of all Operator Workstations/NCMs/printers
n1node	NCM and Operator Workstation records
passwd	passwords
printer	printer definitions
system	system names

NETWORKn

The NETWORKn directory is the 8-character name (no .DOB extension) given on the NET keyword line. The "global download" command is used to copy the NETWORKn.DOB files to the NETWORKn directory for use as Online global databases.

NETWORKn (the following figure) divides into several subdirectories, including DEVICES, which contain the individual NC Files for the appropriate NCMs.

Note: In Figure 5, the .db extension is a shorthand to signify .dbf, .ndx, and other extensions for that file. The name inside the brackets (e.g., [SYS3]) will be the actual name provided in the DDL source file.



* Only present on an OWS that is the archive PC for an Access NC.

Figure 5: C:\JOBXYZ\NETWORKn Directory

The following items are found in this directory:

- operational global databases

  The Operator Workstation features use the global databases located in this directory. These are the operational copies of the same archive databases found in NETWORKn.DOB. Either you place them here by performing a global download from the files in NETWORKn.DOB, or the workstation places them here by synchronizing its database with another node on the N1 network.

  DDL has no part in creating these files other than the fact that they can be uploaded from the NETWORKn.DOB directory, which DDL created.

  Also included is a password file (which the DDL compiler does not generate, but will put here if none exists when a network is created) and a map database built dynamically by the workstation software from the groups and systems.

- DEVICES directory

  The DEVICES directory contains the PCn directory and the NCn directories, which define subdirectories for each workstation and NCM on this network. The compiler builds this directory when compiling the NET.DDL File (at the same time as it builds the NETWORKn directory). Within the NC subdirectory is the downloadable database for that NCM. (See Figure 6.)

  Note: Information inside the brackets (e.g., [object]) represents the actual name defined by the user when generating the database, whether using DDL, GPL, or online generation.

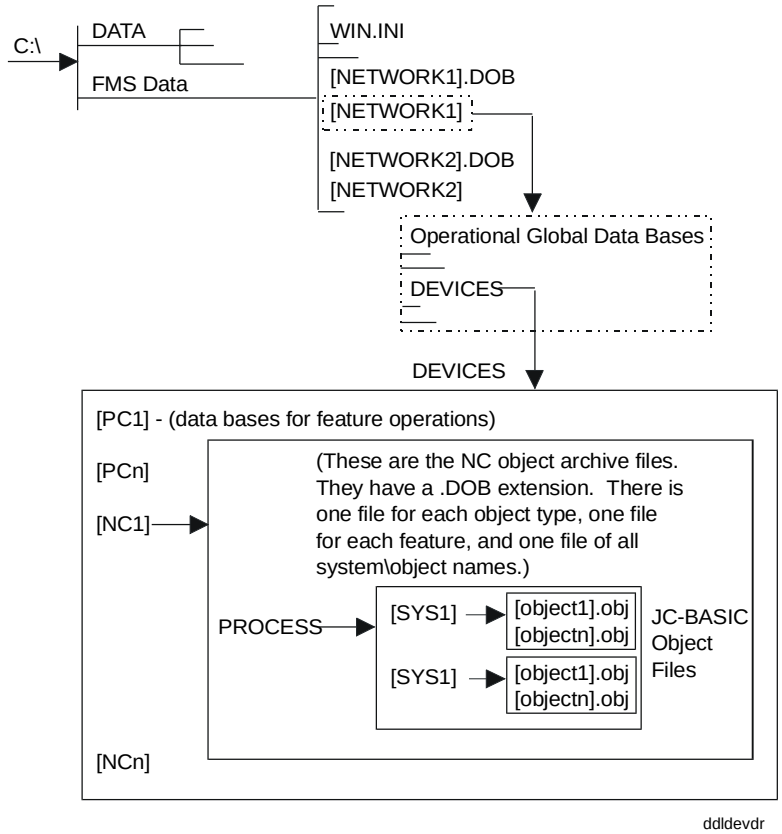


Figure 6: DEVICES Directory

The **PCn** directory name is the 8-character name given on the PC keyword line. This directory contains several databases necessary for Operator Workstation feature operation. The compiler builds one PCn directory for each Operator Workstation in the Global File when compiling the Global File for this network. DDL creates a directory for each workstation, but has no involvement with any of the files contained in these directories.

The compiler creates the **NCn** directory when compiling the Global File. The NCn directory name is the 8-character name given on the NC keyword line. The DDL compiler creates one NCn directory for each NCM in the Global File. The NCn directory contains all of the downloadable databases that will be sent to the NCM and its associated devices. The NCn directory includes the NC databases and PROCESS subdirectory.

DEVICES\NC Databases

The NC databases are a set of data files in the NCn directory that are downloaded to the NCM. The compiler creates these files when compiling the NCM's DDL file for one NCM. These files are the actual individual databases containing the data that is finally downloaded to the field. One database exists for each object type and each feature. Valid object and feature database types are listed below:

ACCD	Access Card
ACCT	Access Controller
ACM	Accumulator objects
ACRD	Access Reader
ACTI	Access Time Zone
AD	Analog Data objects
AI	Analog Input objects
ALARM	Alarm messages
AOD	Analog Data objects
AOS	Analog Output Setpoint objects
BD	Binary Data objects
BI	Binary Input objects
BO	Binary Output objects
C210A/C260A	Control Sequence software objects
C260X/C500X	Control Sequence software objects
CCO	PID control loops
CSG	Generic Control System objects
DCDR	LCP, DC9100, DR9100, TC9100, DX9100, DX91ECH, XT9100, XTM hardware objects
DCM140	DCM140 hardware objects
DLLRGRPS	DL/LR Load Groups
DSC	C210 and C260 hardware objects
DSC8500	DSC8500 hardware objects
FIREOBJ	FIRE controller hardware objects
FIREZONE	Fire Zone object
FPU	FPU controller hardware objects
JCB	JC-BASIC process directory

LCD	Lighting Controller device hardware objects
LCG	Lighting Control Groups
LON	LON hardware objects (LONTCU*)
MC	Multiple Command object
MSD/MSI/MSO	Multi-state objects
N2O	AHU, MIG, NDM, UNT, VAV, PHX, VMA, and VND hardware objects
NCCALSCH	Calendar Schedules
NCWKSCHD	Weekly Schedules
OBJENT	Object Directories
PID	DCM hardware objects
SUBORDINATE	Subordinate for an MC
TOTCONFG	Totalization
TRNCONFG	Trend
XM	XM hardware objects

* For other LON devices, see *LONWORKS Devices Supported by LON NCM Technical Bulletin*.

DEVICES\NC Databases\PROCESS Subdirectory

The NC directory contains a subtree of process object files. (See Figure 6.) The root of this tree is called PROCESS, containing a set of directories named by system names. One directory exists for each system that has a process in it on this NCM. Under each system directory is a set of files, which are the individual process object files named the same as the process object name.

Compiling the NC File creates the PROCESS directory if it is not already there. The JC-BASIC compiler creates a system subdirectory when it adds the first process to that system. The process object files are the output of the JC-BASIC compiler.

- SYSTEMS Directory

  The standard directory named SYSTEMS contains directories for all systems that have not been compiled on this workstation. These systems may have belonged to groups on other workstations; however, because a group hierarchy is valid only for the workstation where it is compiled, the systems are assigned to the SYSTEMS directory when new to a workstation.

- Group and System Directories

  Other sets of subdirectories under the NETWORKn directory (the figure below) are group and system name directories. These directories are a DOS hierarchy created for each group or system.

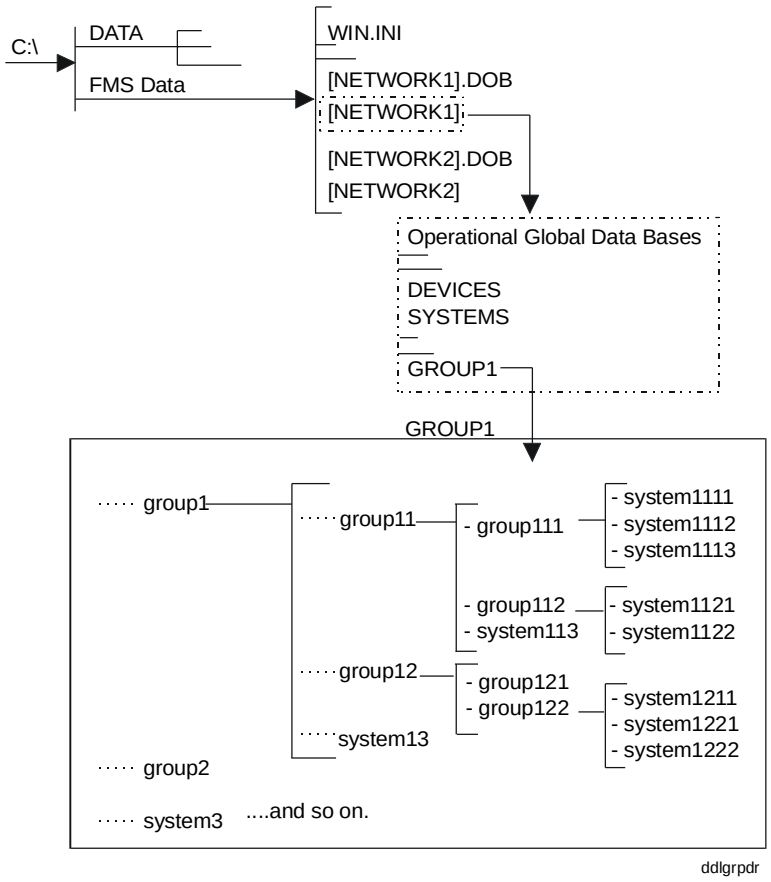


Figure 7: Group and System Directory

The hierarchy supports up to three levels of group names.

The group directories and system directories are created when you compile the Global File that contains them.

- GPL Directory

  The GPL directory is directly under the NETWORKn level. The GPL directory contains GPL source files for that network. (See Figure 8.)

  The DDL compiler automatically creates the GPL directory at the same time it creates the NETWORKn directory. The GPL Editor creates the GPL source files. GPL source files can also be generated by CAE.

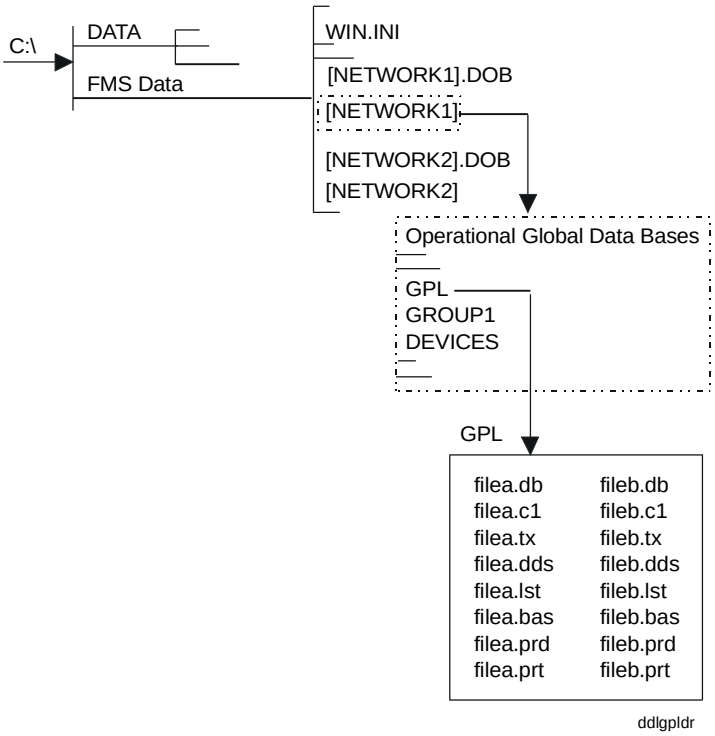


Figure 8: GPL Directory

- COS Directory

  The compiler creates the COS directory at the same time and at the same level as the DEVICES directory. The COS directory contains files of reports sent to this Operator Workstation when the workstation name is the destination for those reports. The Report Router software creates any files in this directory.

- Report Destinations Directories [RPTn]

  The Report Router creates files in this directory, containing reports sent to the current Operator Workstation and having “PC File” as their destination. Each subdirectory name is the same as the destination file name.

DDL Programmer's Manual

Reference

Description of File Tables **Page 3**

- *Column Headings* *3*

Workstation Network/Port Configuration File Syntax 6

- *Workstation File @NET Keyword* *6*
- *Workstation File NET Keyword* *7*
- *Workstation File Port Keyword* *9*

Global File Syntax **10**

- *Global File @GLOBAL Keyword* *10*
- *Global File DEFDES Keyword* *11*
- *Global File GRP (PC Group) Keyword* *12*
- *Global File NC Keyword* *13*
- *Global File PC Keyword* *16*
- *Global File PTR Keyword* *19*
- *Global File RPT Keyword* *21*
- *Global File SYS Keyword* *23*

Model File Syntax **24**

- *Model File @MODEL Keyword* *24*
- *Model File CSMODEL Keyword* *25*

NC File Syntax **33**

- *NC File @NC Keyword* *33*
- *NC File ACM Keyword* *34*
- *NC File AD Keyword* *39*
- *NC File AI Keyword* *42*
- *NC File AOD Keyword* *53*
- *NC File AOS Keyword* *55*
- *NC File BD Keyword* *60*
- *NC File BI Keyword* *63*

© July 1, 2003 Johnson Controls, Inc.

NC File Syntax (Cont.)	Page
• *NC File BO Keyword*	*69*
• *NC File C210A Keyword*	*75*
• *NC File C260A Keyword*	*77*
• *NC File CARD Keyword*	*79*
• *NC File CS Keyword*	*81*
• *NC File D600 Keyword*	*83*
• *NC File DCDR Keyword*	*85*
• *NC File DCM140 Keyword*	*87*
• *NC File DCM Keyword*	*88*
• *NC File DELCARD Keyword*	*89*
• *NC File DELETE Keyword*	*90*
• *NC File DELSUBORDINATE Keyword*	*91* *91*
• *NC File DELTZ Keyword*	*92*
• *NC File DLLR Keyword*	*95*
• *NC File DSC Keyword*	*96*
• *NC File DSC8500 Keyword*	*97*
• *NC File FIRE Keyword*	*99*
• *NC File FPU Keyword*	*100*
• *NC File JCB Keyword*	*101*
• *NC File LCD Keyword*	*102*
• *NC File LCG Keyword*	*106*
• *NC File LON Keyword*	*108*
• *NC File MC Keyword*	*112*
• *NC File MSD Keyword*	*116*
• *NC File MSI Keyword*	*122*
• *NC File MSO Keyword*	*131*
• *NC File N2OPEN Keyword*	*132*
• *NC File PIDL Keyword*	*137*
• *NC File READER Keyword*	*140*
• *NC File SUBORDINATE Keyword*	*152*
• *NC File TIMEZONE Keyword*	*155*
• *NC File XM Keyword*	*157*
• *NC File ZONE Keyword*	

Reference

This *Reference* section contains details of the syntax of the DDL source files. There are four kinds of source files:

- Workstation Network/Port Configuration file
- Global file
- Model file
- NC file

Description of File Tables

All Data Definition Language (DDL) files are constructed within an identical format by entering keywords and parameters. In the tables in this *Reference* section, the keywords appear in alphabetical order within their file type (NET, GLOBAL, MODEL, or NC File types).

Column Headings

This document specifies keywords and parameters under column headings as shown below:

Keyword	KWOPT	R/O/C	Parameter	Type	Restrict	Def	Note
DEFDES		R	original dest. type	integer	1-2		a

The parameters are entered one after another, separated by commas, not on individual lines. (This document presents individual lines for clarity.)

The use and syntax for each of the column headings are as follows:

Keyword

A keyword is the first item on a DDL line. It must be an exact match to the keywords resident on the system. The first keyword listed for each object is its main keyword, which must be entered before entering any subkeywords. You may enter subkeywords in any order after the main keyword. Keywords and subkeywords are case insensitive. (You can use upper or lower case.)

KWOPT

Subkeyword Optional: O (optional) in this column means the subkeyword on this line is optional for its main keyword. If you do not enter the optional subkeyword, the compiler sets each field that the subkeyword defines to the default value listed in the Def (default) column.

R (required) in this column means the subkeyword on this line is required for its main keyword. Failure to provide this subkeyword produces an error.

R/O/C

- R in this column means the parameter on this line is required.
- O in this column means the parameter on this line is optional. For an optional parameter, you may leave the field blank, which results in the compiler using the default value shown in the Def (default) column.
- CR or CO in this column means the parameter on this line is conditional, such that if certain conditions are met, the parameter is either optional (CO) or required (CR). The Note column marker indicates which note defines the condition.

Parameter

Parameter types and definitions are fully described in the *Parameters* section of the *Source File Development* document *(LIT-630020)*.

Restrict	Restrictions. Defines restrictions for the parameter.
Def	[ ] contains the default value for optional parameters. The system uses the default value if you leave an optional parameter blank or if you do not use an optional subkeyword.
Note	Refers you to the note that contains more information about this item.

Purpose of Examples

An example using the keywords and subkeywords follows each syntax description. These examples demonstrate how to use the syntax but are not intended to represent an actual system. Individual examples do not fit together to create an example that compiles without errors.

An example of all three source files defining a complete system is shown in the last part of the *Source File Development* section.

Semantic Rules

The semantic rules describe improper use of the syntax.

Some semantic rules listed in this description are followed by (WARNING). If the compiler finds a violation of a rule that is followed by (WARNING), it generates a warning, but the compile process continues.

If the compiler finds a violation of any other rule, it generates an error; the compile process continues to the end of the file, checking for more errors, but it leaves the database unchanged.

Continuation Character

When entering parameters, a backslash (\) is a "continuation character", indicating that the next parameter for the keyword is on the following input line. The backslash is used (1) before or after a comma or (2) between a keyword (or a subkeyword) and its first parameter.

Workstation Network/Port Configuration File Syntax

Workstation File @NET Keyword

Keyword	KWOPT	R/O/C	Parameter	Type	Restrict	Def	Note
@NET	R		no parameters				
-or-							
@NET +	R						

The first non-comment line in a Workstation Network/Port Configuration file must be the following:

@NET

or

@NET + (Adding "+" indicates an incremental compile.)

Example

```
@NET
PORT "LPT1", 3
NET "XYZ-BLDG","XYZ-BLDG NETWORK","PC1"
N1DIRECT 1,101
```

Note: A complete Workstation/Port File is illustrated at the end of the *Source File Development* document *(LIT-630020)*.

Semantic Rules

@NET (or @NET +)

Must be the first non-comment, non-blank line in the file.

Workstation File
NET Keyword

N1 Accessible Network Definition Syntax							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
NET							
		R	network name	name			
		O	description	char (24)		[null]	
		O	Oper. Work. name	name		[null]	a, b
		O	Duty/Standby	Boolean	Y/N	[N]	f
Choose one:	N1DIRECT						
		R	OWS-Subnet address	integer	1-254		
		R	OWS-Node address	integer	1-254		
		O	UDP Port address	long	2050-65535		e
	NCDIAL						
		O	dial type	char (1)	T, P, I	[T]	c
		R	phone number	phone # (19)			d
	NCDIRECT		no parameters				

Notes:

a For a configured N1DIRECT connection type, the Operator Workstation (OWS) name is required. Also, the subnet and node address for the Operator Workstation must match the subnet and node address as defined in the Global File with the Personal Computer (PC) keyword. You may **not** have an unconfigured workstation if you choose N1DIRECT. For NCDIAL and NCDIRECT networks, if you intend for the Operator Workstation to be a **configured** device, the Operator Workstation name is required and must match the name of the Operator Workstation as defined in the Global File with the PC keyword. A null (blank) Operator Workstation name means this workstation functions as an **unconfigured** device on this network (see Note b).

b Carefully consider your security needs when a person with an unconfigured workstation has access to a network. A security risk may arise, because an unconfigured direct or dial-up connection allows any person who has knowledge of the Metasys® network's name and default Level 1 password to view and edit the password database. Refer to the *Design Considerations: Configured or Non-configured Workstation* section of the *Operator Workstation Technical Bulletin (LIT-636013)* for details.

c T = tone, P = pulse, I = Integrated Services Digital Network (ISDN)

d Enter 0 to use phone number stored in modem by command AT&Z0 = (phone number).
Enter 1 to use phone number stored in modem by command AT&Z1 = (phone number).
Enter 2 to use phone number stored in modem by command AT&Z2 = (phone number).
Enter 3 to use phone number stored in modem by command AT&Z3 = (phone number).

e The User Datagram Protocol (UDP) Port Address parameter is not required unless multiple N1DIRECT entries are present. If multiple N1DIRECT entries are present, each must have a UDP Port Address specified. Multiple N1DIRECT entries are allowed only on Metasys M5 Multinetwork Workstations.

f Duty/Standby is enabled on a per network basis.

Example (single network)

```
NET “XYZ-BLDG”,“XYZ BLDG NETWORK”,“PC1”
N1DIRECT 1,110
```

Example (multiple networks)

```
NET “XYZ-BLDG”,“XYZ BLDG NETWORK”,“PC1”
N1DIRECT1.110
NET “ABC-BLDG”,“ABC BLDG NETWORK”,“PC1”,Y
NCDIAL “T”,“555-1234”
```

Note: ABC-BLDG is a Duty/Standby network.

Semantic Rules

NET	The network name must be unique.
N1DIRECT	The Operator Workstation name must be defined on the NET keyword line. The subnet address should be 1 for standard Metasys systems. Subnets 2-254 are reserved for use with the Metasys router. The node address must be unique. We suggest that N1DIRECT Operator Workstation node addresses be assigned in the range of 100-254 (WARNING).
NCDIRECT	A port must be specified (with the PORT keyword) for the OWS to be able to communicate with the Metasys Network.
NCDIAL	A port must be specified (with the PORT keyword) for the OWS to be able to communicate with the Metasys Network. If dial type is T or P, the port specified with the PORT keyword must be of type NC dial (1). If dial type is I, the port specified with the PORT keyword must be of type NC dial ISDN (5).

Operator Workstation Port Definition Syntax							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
PORT							
		R	port name	char (4)	COM1 COM2 COM3 COM4 LPT1 LPT2		
		O	port use	integer	1-5	[3]	a
		CO	baud	integer	c	[9600]	b

Notes: a Valid port uses are:
1 = NC dial (modem)
2 = NC direct
3 = printer
4 = mouse
5 = NC dial (ISDN)

b Ignored unless the port name is COM1, COM2, COM3, or COM4.

c Valid values are 1200, 2400, 4800, 9600, 19200, 28800, 38400, 57600.

Example

```
PORT "LPT1", 3
PORT "COM1", 1, 2400
```

Semantic Rules

PORT

If the name is LPT1 or LPT2, then the port use must be a printer (3).

If the port name is COM3 or COM4, the port use must be NC direct (2).

Each port name can be used only once for each Operator Workstation.

19200 and higher bauds are valid only if port use is NC Dial modem (1), NC direct (2), or NC dial ISDN (5).

If port use = 1 (NC dial modem), the port baud must be the same as other NC dial (modem) ports. All NC dial (modem) bauds must be the same.

Global File Syntax

Global File @GLOBAL Keyword

Keyword	KWOPT	R/O/C	Parameter	Type	Restrict	Def	Note
@GLOBAL or @GLOBAL +		R	network name	name			

The first non-comment line in the Global file must be the following:

@GLOBAL “network name”

or

@GLOBAL + “network name”
to indicate an incremental compile

Example

```
@GLOBAL "XYZ-BLDG"
```

Note: A complete Global file is illustrated at the end of the *Source File Development* document *(LIT-630020)*.

Semantic Rules

@GLOBAL Must be the first non-comment, non-blank line in the file.

Before using this keyword, you must create the network name by compiling this workstation’s Workstation File (@NET).

-or-

@GLOBAL +

The @GLOBAL + keyword (with + appended) indicates an incremental compile.

Default Destination Definition Syntax							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DEFDES							
		R	original destination type	integer	1-2		a
		R	original device name	name			
		CR	original file name	DOS file			b
		R	default destination type	integer	1-2		a
		R	default device name	name			
		CR	default file name	DOS file			b

Notes: a Valid destination types are:
1 = Operator Workstation or printer
2 = Operator Workstation file

b If the destination type is an Operator Workstation file, this parameter is required.

Example

```
DEFDES 1,"PC2", , 1,"PC1"
```

Semantic Rules

DEFDES The original device name must already exist, and it must be an Operator Workstation or printer.

The default device name must already exist, and it must be an Operator Workstation or printer.

If you select an Operator Workstation file as the original or default destination type, the corresponding device name must identify a previously defined workstation.

Original and default destination files must already exist as report destinations.

Note: A direct connect OWS default destination does not work, but a direct connect OWS can be a default destination.

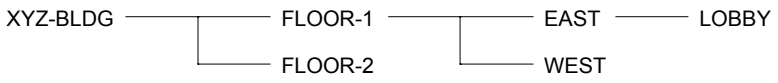
Global File
GRP (PC Group)
Keyword

PC Group Keyword							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
GRP							
		R	group name	name			
		O	group description	char (24)		[null]	
		O	parent group	char (18)		[null]	a
Note: a Use a backslash (\) to separate the Level 1 and Level 2 groups.							

Example

```
GRP “FLOOR-1”, “FIRST FLOOR”
GRP “FLOOR-2”, “SECOND FLOOR”
GRP “EAST”, “EAST AREA”, “FLOOR-1”
GRP “WEST”, “WEST AREA”, “FLOOR-1”
GRP “LOBBY”, “MAIN LOBBY”, “FLOOR-1\EAST”
```

This example would build the network map shown below:



Semantic Rules

GRP

You can use a group name only once in its parent group.

The parent group must already exist and must be no more than two levels deep.

A group cannot have the same name and parent group as a system.

Global File
NC Keyword

Network Control Module (NCM) Definition Syntax							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
NC							
		R	NCM name	name			
		O	NCM description	char (24)		[null]	
		O	graphic symbol no.	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
		R	subnet address	integer	1-254		
		R	node address	integer	1-254		
		O	port 1 type	char (4)	N2/L2/S2/ JC85/“ ”	[N2]	a
		O	port 2 type	char (2)	N2/N2E/L2/ S2/JC85/ ABDH/LON/“ ”	[ ]	a
		O	NCM text language	DOS file			b
		O	NT baud	integer	c	[9600]	d
		CO	S2/JC85/ABDH baud	integer	e	[9600]	f
	IP						g
		R	first octet	integer	0-255		
		R	second octet	integer	0-255		
		R	third octet	integer	0-255		
		R	fourth octet	integer	0-255		

Notes:

a “ ” represents a space.

b The default for the NCM text language comes from the Language variable name in the WIN.INI file.

c Valid bauds are 300, 600, 1200, 2400, 4800, 9600.

d For an NCM200\300\350 defined (via NCSETUP for Windows® software) as an Operator Terminal (OT) type NCM, this baud refers to the OT device baud and not the Network Terminal (NT) device baud.

e Valid bauds are 300, 600, 1200, 2400, 4800, 9600, 19,200. For JC85 port type, 300 and 600 are invalid. For ABDH port type, only 9600 and 19,200 are valid.

f Ignored unless port type is S2, JC85, or ABDH.

g Internet Protocol (IP) subkey is used only when Metasys nodes connect to an IP network (not checked by DDL compiler). If **any** Metasys node (including an Ethernet router) connects to an IP network, **all** N1 nodes require the IP subkey (not checked by DDL compiler).

Example

```
NC “NC-1”", “NC-1 4th Floor”,0,0,1,1,“N2”
NC “NC-2”, “NC-2 20th Floor”,0,0,1,2
NC “NC-3”, “NC-3 36th Floor”,0,0,1,3, “ ”, “N2”
IP 192,9,205,40
```

Semantic Rules

NC The NCM name must be unique (no other workstation, NCM, or printer by that name).

Port type restrictions depend on the type of NCM:

Port Type	NCM101, NCM102	NCM401	NCM200[1]	NCM300 NCM350
N2[3]	Port 1 or Port 2	Port 1 or Port 2	Port 1 and/or Port 2[2]	Port 1 and/or Port 2
N2E	N/A	N/A	N/A	Port 2
L2	Port 1 or Port 2	Port 1 or Port 2	Port 2	Port 2
S2	N/A	Port 1 or Port 2	Port 2	Port 2
JC85	N/A	Port 1 or Port 2	Port 2	Port 2
ABDH	N/A	N/A	Port 2	Port 2
LON	N/A	N/A	N/A	Port 2

1 To support the S2, JC85, or ABDH port types, the NCM200 must contain a 200 Series Network Identity Module (NIM).

2 See the *N2 Communications Bus Technical Bulletin (LIT-636018)* for engineering guidelines for having two N2 trunks on the NCM200 or NCM300.

3 When configuring a Fire Net port, define it as an N2 port in DDL.

For S2 Migration, Port 1 type or Port 2 type is S2, and other port is blank.

For JC/85 Gateway, Port 1 type or Port 2 type is JC85, and other port is blank.

For ALLEN-BRADLEY® Data Highway, Port 2 type is ABDH, and Port 1 type is N2 or blank.

For Echelon® N2 Bus, Port 2 type is N2E, and Port 1 type is N2 or blank.

For a standard NCM, the port type may be L2, N2, or one of each.

For LONWORKS® software, Port 2 type is LONWORKS, and Port 1 type must be blank.

For an NCM200/NCM300/NCM350, both Ports 1 and 2 may be defined as N2. However, DDL does not check to determine if the NCM is one of these types. See the *Dual N2 Bus Application Note (LIT-6363145)* for engineering guidelines on using two N2 trunks on an NCM200/NCM300/NCM350.

For an NCM300/NCM350, Port 1 can be N2 and Port 2 can be N2E. For more details on the Echelon N2 Bus, refer to the *LONWORKS N2E Bus Technical Bulletin (LIT-6364100).*

The subnet address should be 1 for standard Metasys systems. Subnets 2-254 are reserved for use with Metasys router or IP network connections.

If IP is defined, each Metasys software connection to the IP network requires a unique subnet address.

The subnet address must be unique for every NCM or PC connected directly to the Ethernet, including Ethernet routers. The ARCNET® NCMs and PCs attached to an Ethernet router must have the same subnet address as the Ethernet router.

It is possible to have one OWS connected to the IP network and the remaining Metasys nodes connected to ARCNET software and an Ethernet router. In this case, all N1 nodes still require the IP subkey. The rule is if any node requires it, all N1 nodes require it.

The node address must be unique.

A port must not be used by a workstation or printer and a trunk at the same time.

The compiler issues a warning if an NC node address is higher than any PC node address (WARNING).

We suggest you do **not** use GLOBAL as the NCM name. Though this would not result in any performance or system problems, it would create problems if the database is ever decompiled. Specifically, because the decompiler uses GLOBAL.UND as the **fixed** name for the @GLOBAL file, the decompiled @GLOBAL file with the name GLOBAL.UND would overwrite the decompiled @NC file with the name GLOBAL.UND, or vice versa (WARNING).

Global File
PC Keyword

Operator Workstation Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
PC							
		R	Op. Work. name	name			
		O	Op. Work. description	char (24)		[null]	
		O	graphic symbol no.	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
Choose one:	N1DIRECT						
		R	subnet address	integer	1-254		
		R	node address	integer	1-254		
	NCDIAL						
		R	NCM name	name			
		R	NCM port	integer	2, 3, 5, 6		a
		O	baud	integer	b	[1200]	c
		R	primary phone	phone # (19)			d
		O	secondary phone	phone # (19)		[null]	d
		O	number of retries	integer	0-25	[5]	
		O	retry interval in minutes	integer	1-15	[1]	
		O	dial type	char (1)	T or P	[T]	e
	NCDIRECT						
		R	NCM name	name			
		R	NCM port	integer	2-3		
		O	baud	integer	a	[9600]	

Continued on next page . . .

Operator Workstation Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
IP	O						f
		R	first octet	integer	0-255		g
		R	second octet	integer	0-255		
		R	third octet	integer	0-255		
		R	fourth octet	integer	0-255		

Notes:

a Ports 3, 5, and 6 are valid for NCM300/NCM350 only - this is not checked by DDL compiler.

b Valid values are 1200, 2400, 4800, 9600, 19,200, 28,800, 38,400, 57,600 (28800, 38400, 57600 are supported on NCM300/NCM350 Ports 5 and 6 only).

c Baud for NC dial (not ISDN) must be the same as that specified in the NET file, Port keyword, for the NC dial baud.

d Enter 0 to specify use of phone number stored in modem by AT&Z0 = (phone number).

Enter 1 to specify use of phone number stored in modem by AT&Z1 = (phone number).

Enter 2 to specify use of phone number stored in modem by AT&Z2 = (phone number).

Enter 3 to specify use of phone number stored in modem by AT&Z3 = (phone number).

e If port is connected to an ISDN terminal adapter, the dial type field is ignored.

f IP subkey valid with N1 Direct PC only. IP subkey is used only when Metasys nodes connect to an IP network (not checked by DDL compiler). If **any** Metasys node (including an Ethernet router) connects to an IP network, **all** N1 nodes require the IP subkey.

g Octets in an IP address are labeled from left to right. For example, in the address 159,222,20,120, 159 is the first octet, 222 is the second octet, etc.

Example

```
PC “PC1”,“Lobby Operators PC”,0,0
N1DIRECT 1,105
PC “PC2”, “Maintenance Office”,0,0
NCDIAL “NC5”,2,,“555-1234”,,,,“P”
```

Semantic Rules

PC — The OWS name must be unique (no other workstation, NCM, or printer by that name).

N1DIRECT The subnet address should be 1 for standard ARCNET Metasys systems. Subnets 2-254 are reserved for use with Metasys router or IP network connections.

If IP is defined, each Metasys software connection to the IP network requires a unique subnet address.

The subnet address must be unique for every NCM or PC connected directly to the Ethernet, including Ethernet routers. The ARCNET NCMs and PCs attached to an Ethernet router must have the same subnet address as the Ethernet router.

It is possible to have one OWS connected to the IP network and the remaining Metasys nodes connected to ARCNET and an Ethernet router. In this case, all N1 nodes still require the IP subkey. The rule is "if any node requires it, all N1 nodes require it."

The node address must be unique. We suggest assigning N1DIRECT Operator Workstation node addresses in the range of 100-254. The compiler issues a warning if a node address is below 100; however, the compile completes without errors (WARNING).

NCDIAL The NCM name must exist and be an NCM. The NCM port must not be used as N2/L2/S2/JC85/ABDH trunk.

The NCM port must not already be used by another PC or a printer.

For Ports 3, 5, and 6, the NCM must be an NCM300 or NCM350 (no check in DDL).

Port 5 cannot be used with an ARCNET card in the NCM (WARNING).

All NC dial (modem) bauds on the network must be the same. This does not apply to ISDN.

The compiler issues a warning if a PC node address is lower than any NC node address (WARNING).

NCDIRECT The NCM name must already exist and be an NCM.

If the NCM port is 2, it must not be used as N2/N2E/L2/S2/JC85/ABDH trunk.

The NCM port must not already be used by another PC or a printer.

For Ports 3, 5, and 6, the NCM must be an NCM300 or NCM350. (There is no check in DDL.)

Global File
PTR Keyword

Printer Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
PTR							
		R	printer name	name			
		O	printer description	char (24)		[null]	
		O	graphic symbol no.	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
Choose one:	PCDIRECT						
		R	Oper. Work. name	name			
		R	Oper. Work. port	char (4)	LPT1 LPT2 COM1 COM2		
		CO	baud	integer	a	[9600]	b
		O	driver name	DOS file		c	
		O	printer type	char (25)		d	
	NCDIRECT						
		R	NCM name	name			
		R	NCM port	integer	2, 3, 5, 6		e
		O	baud	integer	a	[9600]	
	NCDIAL						
		R	NCM name	name			
		R	NCM port	integer	2, 3, 5, 6		e
		O	baud	integer	a	[1200]	
Continued on next page . . .							

Printer Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		R	primary phone #	phone # (19)			f
		O	secondary phone #	phone # (19)		[null]	f
		O	number of retries	integer	0-25	[5]	
		O	retry interval in minutes	integer	1-15	[1]	
		O	dial type	char (1)	T/P	[T]	

Notes:
a Valid bauds are 300, 600, 1200, 2400, 4800, 9600, 19,200.
b Only use if the port is COM1 or COM2.
c PROPRINT
d IBM® Proprinter®
e Ports 3, 5, and 6 are valid for NCM300/NCM350 only (not checked by DDL compiler).
f Enter 0 to specify use of phone number stored in modem by AT&Z0 = (phone number).
Enter 1 to specify use of phone number stored in modem by AT&Z1 = (phone number).
Enter 2 to specify use of phone number stored in modem by AT&Z2 = (phone number).
Enter 3 to specify use of phone number stored in modem by AT&Z3 = (phone number).

Example

```
PTR “LPTR1”,“Lobby Printer”,0,0
PCDIRECT “PC1”, “LPT1”
PTR “LPTR2”,“Maintenance Printer”,0,0
NCDIRECT “NC1”,3
PTR “LPTR3”, “Backup Printer”,0,0
NCDIAL “NC3”,2,2400, “555-5432”,4,1,“P”
```

Semantic Rules

PTR — The printer name must be unique (no other workstation, NCM, or printer by that name).

PCDIRECT — The Operator Workstation name must already exist.

The port must not be used by any other device.

NCDIRECT The NCM must already exist.

If the NCM port is 2, it must not be used as an L2/N2/N2E/S2/JC85/ABDH trunk.

The NCM port must not already be used by another Operator Workstation or printer.

For Ports 3, 5, and 6, the NCM must be an NCM300 or NCM350 (no check in DDL).

NCDIAL The NCM must already exist. The NCM port must not be used as an L2/N2/S2/JC85/ABDH trunk.

The NCM port must not already be used by another Operator Workstation or printer.

For Ports 3, 5, and 6, the NCM must be an NCM300 or NCM350 (no check in DDL).

Global File
RPT Keyword

Report Group Definitions

Keyword	KWOPT	R/O/C	Parameter	Type	Restrict	Def	Note
RPT							
		R	report group	integer	1-32		a
		R	report name	name			
DEST							b
		R	report type	report type			c
		R	destination type	integer	1-2		e
		R	device name	name			
		CR	file name	DOS file			d

Notes:
- a Report Group numbers greater than 16 require NCMs with more than 2 MB Random Access Memory (RAM). DDL does not check the NCM size.
- b 0-8 DEST subkeys per report type for Report Groups 1-16 (64 DEST subkeys maximum per report group keyword)
  0-16 DEST subkeys per report type for Report Groups 17-32 (128 DEST subkeys maximum per report group keyword)
- c A report type is one of the following strings enclosed in quotation marks (“ ”): status, followup, critical, trans, history, trend, total, card reader.
- d Required if destination type is a PC file.
- e 1 = PC or printer
  2 = PC file

For more information on Report Groups, see the *Report Router/Alarm Management Technical Bulletin (LIT-636114)*.

Example

```
RPT 1,“HARDWARE”
   DEST “CRITICAL”,1,“PC1”
   DEST “CRITICAL”,1,“PC2”
   DEST “CRITICAL”,1,“LPTR1”
   DEST “CRITICAL”,1,“LPTR2”
   DEST “STATUS”,1,“LPTR1”
   DEST “STATUS”,1,“LPTR2”
   DEST “FOLLOWUP”,2,“PC1”,“FOLUPFIL”
RPT 2,“SECURITY”
   DEST “CRITICAL”,1,“PC1”
   DEST “CRITICAL”,1,“LPTR1”
   DEST “STATUS”,1,“LPTR1”
   DEST “FOLLOWUP”,1,“PC1”
RPT 3,“HVAC”
   DEST “CRITICAL”,1,“PC2”
   DEST “CRITICAL”,1,“LPTR2”
   DEST “STATUS”,1,“LPTR2”
   DEST “FOLLOWUP”,1,“PC2”
```

Semantic Rules

RPT — A report group must be defined only once.

DEST — Valid destination types for each report type are:

- Operator Workstation - critical, followup
- Printer - critical, followup, status, trans, cardreader
- Operator Workstation file - all report types

The device name must already exist and must be an Operator Workstation or printer.

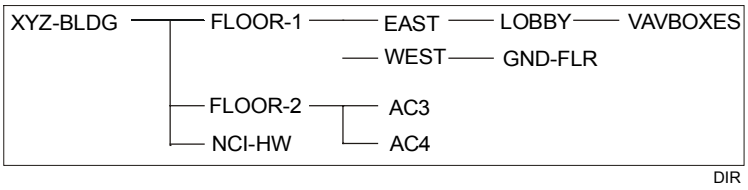
The device name must be an Operator Workstation if the destination type is an Operator Workstation.

System Names Definitions							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
SYS							
		R	system name	name			
		O	system description	char (24)		[null]	
		R	NCM owner name	name			
		R	access report group	integer	1-32		a
		O	parent group	char (27)		[null]	
Note: a Report Group numbers greater than 16 are not supported on NCMs with 2 MB RAM or fewer. DDL does not check the NCM size.							

Example

```
SYS “GND-FLR”, “Ground Floor”,“NC1”,2,\
“FLOOR-1\WEST”
SYS “AC3”,“AC-3-System”,“NC-1”,3,“FLOOR-2”
SYS “AC4”,“AC-4-System”,“NC-1”,3,“FLOOR-2”
SYS “NC1-HW”,“NC-1 Devices”,“NC-1”,1
SYS “VAVBOXES”,“VAV-BOXES”,\
“NC-1”,2,“FLOOR-1\EAST\LOBBY”
```

This example would build the directory structure shown below.



Semantic Rules

SYS — The system name must be unique.

The NCM name must be an existing NCM.

The parent group must already be defined and must be no more than three levels deep.

The access report group must be defined.

System names and PC group names must be unique at the same path level.

Model File Syntax

Model File @MODEL Keyword

Keyword	KWOPT	R/O/C	Parameter	Type	Restrict	Def	Note
@MODEL		R					
or			no parameters				
@MODEL+		R					

The first non-comment line in the MODEL file must be the following:

@MODEL -or-

@MODEL +

(to indicate an incremental compile)

Example

```
@MODEL
CSMODEL "AHU1","AHU"
```

Note: A complete MODEL file is illustrated at the end of the *Source File Development* document *(LIT-630020)*.

Semantic Rules

@MODEL — Must be the first non-comment, non-blank line in the file.

Model File CSMODEL Keyword

Model Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
CSMODEL							
		R	model name	char (8)			
		R	hardware type	char (8)			a
AITITLE	O						
		R	AI group title	char (24)		[null]	
AOTITLE	O						
		R	AO group title	char (24)		[null]	
ADTITLE	O						
		R	AD group title	char (24)		[null]	
BITITLE	O						
		R	BI group title	char (24)		[null]	
BOTITLE	O						
		R	BO group title	char (24)		[null]	
BDTITLE	O						
		R	BD group title	char (24)		[null]	
SPTITLE	O						
		R	SP group title	char (24)		[null]	
MSTITLE	O						
		R	MS group title	char (24)		[null]	
CSAI							b
		R	hardware reference	char (7)			

Notes: a Hardware type must be AHU, LCP, MIG, PHX, UNT, VAV, VMA, VND, NDM, DX9100, DX91ECH, DC9100, DR9100, TC9100, XT9100, XTM, or LONTCU. (For other LONWORKS devices, see *LONWORKS Compatible Devices Supported by NCM350 Technical Bulletin (LIT-1162100).*

b 0-16 CSAI subkeywords are allowed.

Continued on next page . . .

Model Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		O	AI can be overridden.	Boolean	Y/N	[N]	c
		O	AI can be adjusted.	Boolean	Y/N	[N]	d
		R	AI descriptor	char (8)			e
		O	AI units	char (6)			
CSAO							f
		R	hardware reference	char (7)			
		O	AO can be overridden.	Boolean	Y/N	[N]	c
		O	AO can be adjusted.	Boolean	Y/N	[N]	d
		R	AO descriptor	char (8)			e
		O	AO units	char (6)			
CSAD							g
		R	hardware reference	char (7)			
		O	AD can be overridden.	Boolean	Y/N	[N]	c
		O	AD can be adjusted.	Boolean	Y/N	[N]	d
		R	AD descriptor	char (8)			e
		O	AD units	char (6)			
CSBI							h
		R	hardware reference	char (7)			
		O	BI can be overridden.	Boolean	Y/N	[N]	c
		O	BI can be adjusted.	Boolean	Y/N	[N]	d
		R	BI descriptor	char (8)			e

Notes:
- c If Yes, a manual override can be sent to object at Priority 1.
- d If Yes, an MCO, process, Metalink™ software, or manual adjust can adjust object at Priority 2 or 3.
- e Must be non-null and must contain non-blank characters.
- f 0-16 CSAO subkeywords are allowed.
- g 0-32 CSAD subkeywords are allowed.
- h 0-16 CSBI subkeywords are allowed.

Continued on next page . . .

Model Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		R	BI units-Open	char (6)	e		
		R	BI units-Closed	char (6)			e
CSBO							i
		R	hardware reference	char (7)			
		O	BO can be overridden.	Boolean	Y/N	[N]	c
		O	BO can be adjusted.	Boolean	Y/N	[N]	d
		R	BO descriptor	char (8)			e
		R	BO units-Open	char (6)			e
		R	BO units-Closed	char (6)			e
CSBD							j
		R	hardware reference	char (7)			
		O	BD can be overridden.	Boolean	Y/N	[N]	c
		O	BD can be adjusted.	Boolean	Y/N	[N]	d
		R	BD descriptor	char (8)			e
		R	BD units-Open	char (6)			e
		R	BD units-Closed	char (6)			e

Notes:
- c If Yes, a manual override can be sent to object at Priority 1.
- d If Yes, an MCO, process, Metalink software, or manual adjust can adjust object at Priority 2 or 3.
- e Must be non-null and must contain non-blank characters.
- i 0-16 CSBO subkeywords are allowed.
- j 0-32 CSBD subkeywords are allowed.

Continued on next page . . .

Model Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
CSSP							k
		R	hardware reference	char (7)			
		O	SP can be overridden.	Boolean	Y/N	[N]	c
		O	SP can be adjusted.	Boolean	Y/N	[N]	d
		R	SP descriptor	char (8)			e
		O	SP units	char (6)			
CSMS							l
		R	hardware reference	char (7)			
		O	MS can be overridden.	Boolean	Y/N	[N]	c
		O	MS can be adjusted.	Boolean	Y/N	[N]	d
		R	MS descriptor	char (8)			e
		R	MS state 0 text	char (6)		[null]	e
		R	MS state 0 value	integer	-32767 to 32767	[0]	n
		CR	MS state 1 text	char (6)		[null]	m
		O	MS state 1 value	integer	-32767 to 32767	[0]	n

Notes:

c If Yes, a manual override can be sent to object at Priority 1.

d If Yes, an MCO, process, Metalink software, or manual adjust can adjust object at Priority 2 or 3.

e Must be non-null and must contain non-blank characters.

k 0-32 CSSP subkeywords are allowed.

l 0-2 CSMS subkeywords are allowed.

m Required if corresponding value is entered. If required, must be non-null and must contain non-blank character.

n If you are mapping the CSMS attribute to a LONWORKS device, make sure to set each state's value to the number of the state. For example, set the MS state 0 value to 0, MS state 1 value to 1, etc., up to the required number of states. Leave text and value fields for unused states undefined, so these states are hidden in the Control System (CS) object.

Continued on next page . . .

Model Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		CR	MS state 2 text	char (6)		[null]	m
		O	MS state 2 value	integer	-32767 to 32767	[0]	n
		CR	MS state 3 text	char (6)		[null]	m
		O	MS state 3 value	integer	-32767 to 32767	[0]	n
		CR	MS state 4 text	char (6)		[null]	m
		O	MS state 4 value	integer	-32767 to 32767	[0]	n

Notes:
- m Required if corresponding value is entered. If required, must be non-null and must contain non-blank character.
- n If you are mapping the CSMS attribute to a LONWORKS device, make sure to set each state's value to the number of the state. For example, set the MS state 0 value to 0, MS state 1 value to 1, etc., up to the required number of states. Leave text and value fields for unused states undefined, so these states are hidden in the CS object.

Example

```
CSMODEL "AHU-22","AHU"
   AITITLE "ANALOG INPUTS"
   BITITLE "BINARY INPUTS"
   BDTITLE "BINARY DATA"
   CSAI "AI1",N,N,"ZN TMP 2","DEGF"
   CSAI "AI2",N,N,"ZN TMP 3","DEGF"
   CSAI "AI3",N,N,"AT TMP 1","DEGF"
   CSAI "AI4",N,N,"AT TMP 2","DEGF"
   CSAI "AI5",N,N,"DIS HUM","% RH"
   CSBI "BI1",N,N,"LOW STAT","NOR","LOW"
   CSBI "BI2",N,N,"HI STAT","NOR","HIGH"
   CSBD "BD30",Y,Y,"OCCUPIED",\
   "UNOCC","OCC"
```

Note: A complete MODEL file is illustrated at the end of the *Source File Development* document *(LIT-630020)*.

Semantic Rules

See the *Control System (CS) Object Technical Bulletin (LIT-636102)* document for information on which controller point types are valid for the attribute groups.

CSMODEL The model name must be unique.

The hardware type must exist as a hardware model in the hardware model database.

CSAI The hardware reference must be valid for the hardware model and must be controller point type Analog Input (AI), Analog Output (AO), or IF. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSAI keyword is present, the AITITLE cannot be null.

CSAO The hardware reference must be valid for the hardware model and must be controller point type AI, AO, or IF. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSAO keyword is present, the AOTITLE cannot be null.

CSAD The hardware reference must be valid for the hardware model and must be controller point type AI, AO, IBY, IF, or II. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSAD keyword is present, the ADTITLE cannot be null.

CSBI The hardware reference must be valid for the hardware model and must be controller point type BI, BO, IBit, or IByte. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSBI keyword is present, the BITITLE cannot be null.

CSBO The hardware reference must be valid for the hardware model and must be controller point

type BI, BO, IBit, or IByte. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSBO keyword is present, the BOTITLE cannot be null.

CSBD — The hardware reference must be valid for the hardware model and must be controller point type BI, BO, IBit, or IByte. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSBD keyword is present, the BDTITLE cannot be null.

CSSP — The hardware reference must be valid for the hardware model and must be controller point type AI, AO, or IF. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSSP keyword is present, the SPTITLE cannot be null.

CSMS — The hardware reference must be valid for the hardware model and must be controller point type IBy or II. If either the override or adjust flag is set to Yes (Y), the hardware reference must be commandable for the hardware model.

If one CSMS keyword is present, the MSTITLE cannot be null.

IMPORTANT:	Though the following is not checked by the DDL compiler, it is important for you to consider. If you are mapping a CS object attribute and a standard object to the same hardware reference, set both the Override and Adjust flags to No (False) for the CS object attribute. Similarly, if you are mapping more than one CS object attribute to the same hardware reference, make sure only one has the Override flag set to Yes and only one has the Adjust flag set to Yes. This is to ensure that there is only one command path to the hardware reference.

NC File
@NC Keyword

Keyword	KWOPT	R/O/C	Parameter	Type	Restrict	Def	Note
@NC		R	network name	name			
or							
@NC +		R	NCM name	name			

The first non-comment line in the NC file must be the following:

@NC “network name”, “NCMname”

-or-

@NC + “network name”, “NCMname”

(to indicate an incremental compile).

Example

```
@NC “XYZ-BLDG”,“NC1”
```

Note: A complete NC file is illustrated at the end of the *Source File Development* document *(LIT-630020)*.

Semantic Rules

@NC Must be the first non-comment, non-blank line in the file.

Before using this keyword, you must create:

- the network name by compiling this workstation’s workstation file (@NET)
- the NCM name by compiling the Global file (@GLOBAL)

Accumulator Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
ACM							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operating instruction	integer	0-32767	[0]	
Choose one:	XMHW						
		R	XM type	integer	1-3		a
		O	hardware type	integer	1-2	[1]	b
		O	slot number	integer	1-32	[1]	
		CO	debounce filter (msec)	integer	12-3060	[24]	c
		O	LED On when closed?	Boolean	Y/N	[Y]	
	DSCHW						
		O	logical point type	char (3)	TOT	[TOT]	
		O	logical point number	integer	1-255	[1]	
	FPUHW						
		O	slot number	integer	1-16	[1]	

Notes:
a Valid XM types are:
1 = XBN
2 = XRM
3 = XRL or XRE

b Valid hardware types are:
1 = single
2 = Form C

c Use the debounce filter only if the hardware type is single (1). Enter the debounce filter to represent milliseconds (msec). The debounce filter must be a multiple of 12.

Continued on next page . . .

Accumulator Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (options continued from previous page):	DCDRHW						
		R	HW device type	char (8)			d
		R	HW reference	char (7)			d
	LONHW						
		R	HW device type	char (8)			m
		R	HW reference	char (7)			m
	N2OPENHW						
		O	point type	char (3)	BI	[BI]	
		O	point address	integer		[7]	k
RANGE	O						
		O	pulse count	fp		[0.0]	
		O	rate constant	integer	0-2	[2]	e
FILTER	O						
		O	filter weight	fp	>1.0		f
UNITS	O						
		O	engineering units	char (6)		[KW]	
		O	analog consumpt. unit	char (6)		[KWH]	
		O	decimal display position	integer	0-3	[1]	

Notes:

d Valid DCDR **device types** and **hardware references** are:

LCP	none
DX9100	CNT1-8
PMnAC1-8	(n = 1-12)
XTnCNT1-8	(n = 1-8)
DX91ECH	CNT1-8
PMnAC1-8	(n = 1-12)
XTnCNT1-8	(n = 1-8)
DC9100	Total 1-2 (available on DO9100 only)
DR9100	none
TC9100	none
XT9100	CNT1-8
XTM	CNT1-8

e For the rate constant, valid values are:
0 = seconds, 1 = minutes, 2 = hours

f The default parameter for these fields is undefined, which equates to an FFFFFFFF in the database.

k Refer to *Accumulator (ACM) Object Technical Bulletin (LIT-636076)* for mapping restrictions.

m Valid LONHW device types and hardware references are:
LONTCU xxACxxx (The xs are placeholders for specific addressing numbers and characters. See the *LONWORKS Compatible Devices Supported by NCM350 Technical Bulletin [LIT-1162100]*).

Continued on next page . . .

Accumulator Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
ALARMSET	O						
		O	differential	fp	>0.0		g
		O	setpoint	fp			f
		O	normalband	fp	>0.0		f
		O	low alarm limit	fp			f
		O	high alarm limit	fp			f
		O	warning delay (minutes)	integer	0-255	[1]	
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	h
		O	comm. disable	Boolean	Y/N	[N]	
		CO	warning message #	integer	0-255	[0]	i
		CO	alarm message #	integer	0-255	[0]	i
		O	normal report type	integer	0-6	[0]	j
		O	warning report type	integer	0-6	[0]	j
		O	alarm report type	integer	0-6	[0]	j
		O	override report type	integer	0-6	[0]	j

Notes:

f The default parameter for these fields is undefined, which equates to an FFFFFFFF in the database.

g The differential applies only if either a high or low limit is used or if a setpoint is defined.

h Ignored if the point history flag is No.

i Ignored if the corresponding report type is 0.

j Valid report types (0-6) are:

0 = no report

1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4

5 = followup

6 = status

Example

```
ACM “AC3”,“ACM1”
   HARDWARE “NC1-HW”,“XBN1”
   GRAPHICS 1,1
   XMHW 1,1,2,Y
   RANGE 3.5,2
   FILTER 100.0
   UNITS “KW”,“KWH”
   ALARMSET 1.0,50.0,5.0,10.0,100.0,23
   REPORT y,y,y,y,100,100,2,2,2,2
```

Semantic Rules

ACM — The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist, match the selected controller (for example, XM, DCDR, DSC8500, FPU, N2OPEN), and be on the same NCM as the ACM.

ALARMSET

Either you must define both the setpoint and normalband, or neither of them.

Use a differential only if either a high limit, low limit, or setpoint is defined.

If the differential is not defined, use 0 for its value in the following comparisons.

If you define the differential (diff), setpoint (sp), and normalband (nb), then the following must be true:

[sp - (nb/2) + diff] < [sp + (nb/2) - diff].

If the low limit is defined, then:

[low limit + diff] < [sp - (nb/2)].

If the high limit is defined, then:

[sp + (nb/2)] < [high limit - diff].

If both the low and high limits are defined, then:

low limit < high limit.

XMHW	If hardware type is “single,” the slot number must be free. If the hardware type is “Form C,” the slot number and the slot number + 1 must be free; the slot number must be an odd number. If the XM type is 2 or 3, the slots are restricted to 1-8. The debounce filter must be divisible by 12.
DCDRHW	The hardware reference must be valid for the specified hardware device type. The hardware reference must be free. The hardware reference must be valid for an ACM.
LONHW	The hardware reference must be valid for the specified hardware device type. The hardware reference must be free. The hardware reference must be valid for an ACM.
DSCHW	LPN must be free for the logical point type chosen.
FPUHW	Slot number must be free.
N2OPENHW	The point type and point address must be free.

IMPORTANT:	This information applies to objects mapped to DCDRHW and N2OPENHW. Though the following is not checked by the DDL compiler, it is important for you to consider. If you are mapping a CS object attribute and a standard object to the same hardware reference (for N2OPENHW, the hardware reference is the combination of the point type and point address), make sure the Override and Adjust flags are set to No (False) for the CS object attribute. This is to ensure that there is only one command path to the hardware reference.

NC File
AD Keyword

Analog Data Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
AD							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ASSOCINP	O						
		R	associated object type	char (8)		[null]	a
		R	associated system name	name		[null]	f
		R	associated object name	name		[null]	f
		R	associated attribute	char (8)		[null]	f
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
UNITS	O						
		O	engineering units	char (6)		[DEGF]	
		O	decimal display posit.	integer	0-3	[1]	
		O	initial value	fp		[55.0]	

Notes:

a Valid object types are:

ACM	BO	DCM	FPU	MSO
AD	C210A	DCM140	JCB	N2OPEN
AI	C260A	DLLR	LCD	PIDL
AOD	C260X	DSC-1000	LCG	READER
AOS	C500X	DSC8500	LON	MC
BD	CS	D600	MSD	ZONE
BI	DCDR	FIRE	MSI	XM

f For mapping an AD to NCM diagnostics (NCM 200, 300 Series only), associated object type and associated system/object names are ignored; however, they must **not** be blank. Enter the following associated attributes: IDLE, FREEMEM, N1RX, N1TX, JCBTIME, N1RXERR, IPRXERR, STBYSTAT, N2APPS, N2ACPS, N2BPPS, N2BCPS. Diagnostics are obtained from the NCM on which the AD is located.

Continued on next page . . .

Analog Data Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
AD							
FILTER	O						
		O	filter weight	fp	>1.0		b
ALARMSET	O						
		O	differential	fp	>0.0		b
		O	setpoint	fp			b
		O	normalband	fp	>0.0		b
		O	low limit	fp			b
		O	high limit	fp			b
		O	delay time	integer	0-255	[1]	
		O	adjust disabled	Boolean	Y/N	[N]	
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[N]	
		CO	save pt. history flag	Boolean	Y/N	[N]	c
		O	comm. disable	Boolean	Y/N	[N]	
		CO	warning message #	integer	0-255	[0]	d
		CO	alarm message #	integer	0-255	[0]	d
		O	normal report type	integer	0-6	[0]	e
		O	warning report type	integer	0-6	[0]	e
		O	alarm report type	integer	0-6	[0]	e
		O	override report type	integer	0-6	[0]	e

Notes:
- b The default parameter for this field is undefined, which equates to an FFFFFFFF in the database.
- c Ignored if the point history flag is No.
- d Ignored if the corresponding report type is 0.
- e Valid report types (0-6) are:
  - 0 = no report
  - 1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
  - 5 = followup
  - 6 = status

Example 1

```
AD “AC4”,“OAT”
   ASSOCINP “AI”,“AC3”,“OAT”,“VALUE”
   UNITS “DEG C”,0,15.0
```

Example 2

```
AD “N2OPEN1”, “idle”, “ ”
   ASSOCINP “AD”,“NONE”,“NONE”,“IDLE”
   GRAPHICS 0, 0
   UNITS “%”,1,55.00000
   ALARMSET, , , 30.00000, 100.000, 1
   REPORT N, N, , N, , , 0, 0, 0, 0
```

Semantic Rules

AD	The system name must already exist. The object name must be unique. Although not checked by the compiler, if the AD is going to map to an ALLEN-BRADLEY host system, the system name must be a number from 0 to 65535, and the object name must be a number from 0 to 799. Although not checked by the compiler, if the AD is going to map to a JC/85/40 point, there are specific requirements for system and object names. See the *JC/85 Gateway Application Note (LIT-6363147)* for details.
ASSOCINP	The system\object must exist (WARNING). The attribute must be a legal floating point attribute for that object.

ALARMSET

Either define both the setpoint and normalband, or do not define either of them.

Use a differential only if either a high limit, low limit, or setpoint is defined.

If the differential is not defined, use a value of 0 for it in the following comparisons.

If you define the differential (diff), setpoint (sp), and normalband (nb), then:

[sp - (nb/2) + diff] < [sp + (nb/2) - diff].

If the low limit is defined, then:

[low limit + diff] < [sp - (nb/2)].

If the high limit is defined, then:

[sp + (nb/2)] < [high limit - diff].

If both the low and high limits are defined, then:

low limit < high limit.

NC File
AI Keyword

Analog Input Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
AI							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
Continued on next page . . .							

Analog Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (options continued on next page):	DCDRHW						
		R	HW device type	char (8)			a
		R	HW reference	char (7)			a
	DCM140HW						
		O	slot number	integer	1-10	[1]	
		O	analog type	integer	1-4	[1]	b
		O	point type	integer	0-1	[0]	c
		CO	subslot number	integer	1-2	[1]	d
	DCMHW						
		O	slot number	integer	1-10	[1]	
		O	analog type	integer	1-3	[1]	b
	DSCHW						
		O	logical point type	char (3)		[FUL]	e
		O	logical point #	integer	1-255	[1]	
	FPUHW						
		O	slot number	integer	1-16	[1]	
		O	filter tolerance	fp	0.2-100%	[0.2]	

Notes:

a Valid DCDR **device types** and **hardware references** are:

LCP	AI1-8
DX9100	AI1-8, XTnAI1-8 (n = 1-8), CNT1-8, PMnAC1-8 (n = 1-12), XTnCNT1-8 (n = 1-8)
DX91ECH	AI1-8, XTnAI1-8 (n = 1-8), CNT1-8, PMnAC1-8 (n = 1-12), XTnCNT1-8 (n = 1-8)
DC9100	AI1-8, TOTAL1-2
DR9100	AI1-4
TC9100	AI1-4
XT9100	AI1-8, CNT1-8
XTM	AI1-8, CNT1-8

b Valid analog types are:
1 = 1 K ohm
2 = 100 ohm - not available if point type is multiple AI (MAI)
3 = VOLT/AMP
4 = V/A LOW END REL - not available with DCMHW

c 0 = AI, 1 = multiple AI (MAI)

d Subslot ignored unless point type is multiple AI (MAI).

e For DSCHW, valid logical point types are: ADP, ASP, FUL, INC, LTD, RAT, and TOT.

Continued on next page . . .

Analog Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (options continued from previous page)	LONHW						
		R	HW device type	char (8)			q
		R	HW reference	char (7)			q
	N2OPENHW						
		O	point type	char (3)	AI	[AI]	
		O	point address	integer		[1]	p
Choose zero or one	RANGE						f
		O	standard range	integer	1-111	[1]	
	LINEPARM						f
		O	linearization parm. 1	fp		[0.0]	
		O	linearization parm. 2	fp		[0.1]	
		O	linearization parm. 3	fp		[0.2]	
		O	linearization parm. 4	fp		[0.3]	
FILTER	O						g
		O	filter weight	fp	>1.0		h
FLOW	O						i
		O	flow coefficient	fp	>0.0		h
SPANS	O						i
		R	low input span equat.	fp			h
		R	high input span equat.	fp			h
		R	low output span equat.	fp			h
		R	high output span equat.	fp			h

Notes:

f If DCDRHW, DSCHW, LONHW, or N2OPENHW, do not enter RANGE or LINEPARM.

If DCMHW, DCM140HW, or FPUHW, you must use either the RANGE or LINEPARM.

If you use RANGE, the corresponding values in the Linear Parameter Table are used for the linear parameter (parm.) values. (This table follows the AI semantic rules.)

If you use LINEPARM, then the range is set to 0.

g If DCDRHW, FPUHW, DSCHW, LONHW, or N2OPENHW, do not use FILTER subkey.

h The default parameter for this field is undefined, which equates to an FFFFFFFF in the database.

i If DCDRHW, LONHW, or N2OPENHW, do not use FLOW or SPANS subkeys.

p Refer to *Analog Input (AI) Object Technical Bulletin (LIT-636080)* for mapping restrictions.

q Valid LONHW device types and hardware references include LONTCU xxAIxxx, xxAC,xxx. (The xs are place holders for specific addressing numbers and characters. See the device's technical bulletin for details.) For other LONWORKS devices, see *LONWORKS Compatible Devices Supported by NCM350 Technical Bulletin (LIT-1162100)*.

Continued on next page . . .

Analog Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
UNITS	O						
		O	engineering units	char (6)		[DEGF]	
		O	decimal display position	integer	0-3	[1]	j
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
ALARMSET	O						
		O	differential	fp	>0.0		h, k
		O	setpoint	fp			h, k
		O	normalband	fp	>0.0		h, k
		O	low alarm limit	fp			h, k
		O	high alarm limit	fp			h
		O	delay time	integer	0-255	[1]	j
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. hist. flag	Boolean	Y/N	[N]	l
		O	comm. disable	Boolean	Y/N	[N]	
		CO	warning message #	integer	0-255	[0]	m
		CO	alarm message #	integer	0-255	[0]	m
		O	normal report type	integer	0-6	[0]	n
		O	warning report type	integer	0-6	[0]	n
		O	alarm report type	integer	0-6	[0]	n
		O	override report type	integer	0-6	[0]	n, o

Notes:

h The default parameter for this field is undefined, which equates to an FFFFFFFF in the database.

j Ignored for DCDRHW/LONHW if the HW Reference is a DCDR counter item/LON counter network variable. The default value is then set to 0.

k Ignored for DCDRHW/LONHW if the HW Reference is a DCDR counter item/LON counter network variable. The default value is then set to undefined.

l Ignored if the point history flag is No.

m Ignored if the corresponding report type is 0.

n Valid report types (0-6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
5 = followup
6 = status

o If DCDRHW or LONHW, override report type is ignored.

Example

```
AI “AC3”, “ROOMTEMP”
   HARDWARE “NC1-HW”, “DCM1”
   DCMHW 5,1
   RANGE 1
```

Semantic Rules

AI — The system name must already exist. The object name must be unique.

HARDWARE — The system\object must already exist, match the type of selected controller (for example, DCM, DCM140, DSC8500, FPU, N2OPEN, DCDR), and be on the same NCM as the AI.

RANGE — If DCMHW or DCM140HW, then STD Range must be 1 to 25 or 34 to 111. If FPUHW, STD Range must be 26 to 33.

ALARMSET

Note: If the DCDR hardware device type is DR9100, alarm limits may be defined only if the input range defined in the controller is type 0 (that is, 0-100%).

Either define both the setpoint and normalband, or do not define either of them. Use a differential only if a high limit, low limit, or setpoint is defined. If the differential is not defined, use 0 for its value in the following comparisons.

If you define the differential (diff), setpoint (sp), and normalband (nb), then the following must be true:

[sp - (nb/2) + diff] < [sp + (nb/2) - diff].

If the low limit is defined, then the following must be true:

[low limit + diff] < [sp - (nb/2)].

If the high limit is defined, then the following must be true:

[(sp + nb)/2] < [high limit - diff].

If both the low and high limits are defined, then the following must be true:

low limit < high limit.

SPANS Low input must be less than High input. Low output must not equal High output.

DCDRHW The hardware reference must be valid for the specified hardware device type.

The hardware reference must be free.

The hardware reference must be valid for an AI.

DCM140HW

If point type is AI, the slot number must be free. If point type is multiple AI, the slot and subslot must be free.

If point type is multiple AI (MAI), 100 ohm analog type is not allowed.

DCMHW The slot number must be free.

DSCHW Logical Point Number (LPN) must be free for the logical point type chosen.

FPUHW The slot number must be free.

LONHW The hardware reference must be valid for the specified hardware device type.

The hardware reference must be free.

The hardware reference must be valid for an AI.

N2OPENHW

Point type and address must be free.

IMPORTANT:	This information applies to objects mapped to N2OPENHW. Though the following is not checked by the DDL compiler, it is important for you to consider. If you are mapping a CS object attribute and a standard object to the same hardware reference (the hardware reference is the combination of the point type and point address), make sure the Override and Adjust flags are set to No (False) for the CS object attribute. This ensures that there is only one command path to the hardware reference.

Linear Parameter for Standard Ranges

For DCMHW or DCM140HW, use standard ranges 1 to 25 and 34 to 111.
For FPUHW, use standard ranges 26 to 33.

Range	Type	Parm 1	Parm 2	Parm 3	Parm 4
1	DEGF 1000 ohm Ni	-49.929	89.144	-4787.6	221.94
2	DEGF 1000 ohm Pl	-116.32	100.75	979.87	10.941
3	DEGF CPD silicon	-16.936	66.482	-5624.1	414.23
4	DEGF 100 ohm Pl	-49.972	71.417	437.8	3.0646
5	DEGC 1000 ohm Ni	-45.516	49.525	-2659.7	123.3
6	DEGC 1000 ohm Pl	-82.397	55.987	6.2715	543.93
7	DEGC CPD silicon	-27.197	36.979	-3153.7	235.82
8	DEGC 100 ohm Pl	-45.54	39.676	243.26	1.6975
9	WC IDP001 0/0.1	-0.025	0.03656	0.0	0.0
10	WC IDP002 0/0.25	-0.0625	0.09139	0.0	0.0
11	WC IDP005 0/0.5	-0.125	0.18278	0.0	0.0
12	WC IDP010 0/1	-0.25	0.36557	0.0	0.0
13	WC IDP030 0/3	-0.75	1.09670	0.0	0.0
14	WC IDP050 0/5	-1.25	1.82784	0.0	0.0
15	WC IDP100 0/10	-2.5	3.65568	0.0	0.0
16	mBAR IDP001 0/0.249	-0.06228	0.09106	0.0	0.0
17	mBAR IDP002 0/0.623	-0.15576	0.22775	0.0	0.0
18	mBAR IDP005 0/1.245	-0.31125	0.45513	0.0	0.0
19	mBAR IDP010 0/2.491	-0.62275	0.91063	0.0	0.0

Continued on next page . . .

Linear Parameter for Standard Ranges (Cont.)					
For FPUHW, use standard ranges 26 to 33. For DCM140HW and DCMHW, use standard ranges 1 to 25 and 34 to 111.					
Range	**Type**	**Parm 1**	**Parm 2**	**Parm 3**	**Parm 4**
20	mBAR IDP030 0/7.472	-1.8680	2.73152	0.0	0.0
21	mBAR IDP050 0/12.45	-3.1125	4.55132	0.0	0.0
22	mBAR IDP100 0/24.91	-6.2275	9.10629	0.0	0.0
The following ranges display mA or VDC. Use span function to show appropriate engineering units.					
23	4/20 mA input	0.0	5.86081	0.0	0.0
24	0/10 VDC input	0.0	2.92454	0.0	0.0
Note: If you use the AI with a Delta-P sensor to calculate flow, you cannot use ranges 23 or 24; you must range the AI to match the sensor.					
25	W560/HE-6110 0/5 VDC	0.0	58.4908	0.0	0.0
26	ANT101 0 to 100F	-0.03364	52.1918	-1741.36	57.6882
27	ANT102 40 to 140F	39.94782	52.2460	-1783.97	67.5819
28	ANT103 -50 to 150F	-49.98669	111.394	-7642.14	464.659
29	ANT104 -50 to 250F	-50.05439	179.105	-19755.2	1930.92
30	ANT101 -18 to 38C	-17.7962	28.9954	-967.423	32.0491
31	ANT102 4 to 60C	4.41553	29.0255	-991.095	37.5455
32	ANT103 -46 to 66C	-45.5480	61.8858	-4245.63	258.144
33	ANT104 -46 to 121C	-45.5857	99.5000	-10975.1	1072.73
Notes: If you are adding an ANV, ANC, or ANP analog card to an AI software object, you must calculate the linearization parameters. The equations for calculating linearization parameters are in the *Analog Input (AI) Object Technical Bulletin (LIT-636080)*.					
For FPUHW, use standard ranges 26 to 33. For DCM140HW and DCMHW, use standard ranges 1 to 25 and 34 to 111.					
34	WC IDP250 0/25	-6.250000	9.139063	0.0	0.0
35	WC IDPB001 -0.1/0.1	-0.150000	0.073113	0.0	0.0
36	WC IDPB002 -0.25/0.25	-0.375000	0.182781	0.0	0.0
37	WC IDPB005 -0.5/0.5	-0.750000	0.365563	0.0	0.0
38	WC IDPB010 -1.0/1.0	-1.500000	0.731125	0.0	0.0
39	WC IDPBO25 -2.5/2.5	-3.75000	1.827813	0.0	0.0
40	WC IDPB050 -5/5	-7.500000	3.655625	0.0	0.0
41	WC DPT-2640-1 0/0.1	0.0	0.058490	0.0	0.0
42	WC DPT-2640-2 0/0.25	0.0	0.146225	0.0	0.0
43	WC DPT-2640-3 0/0.5	0.0	0.292450	0.0	0.0
44	WC DPT-2640-4 0/1	0.0	0.584900	0.0	0.0
45	WC DPT-2640-5 0/2.5	0.0	1.462250	0.0	0.0
46	WC DPT-2640-6 0/5	0.0	2.924500	0.0	0.0
Continued on next page . . .					

Linear Parameter for Standard Ranges (Cont.)					
For FPUHW, use standard ranges 26 to 33. For DCM140HW and DCMHW, use standard ranges 1 to 25 and 34 to 111.					
Range	**Type**	**Parm 1**	**Parm 2**	**Parm 3**	**Parm 4**
47	WC DPT-2640-7-0/10	0.0	5.849000	0.0	0.0
48	WC DPT-2640-8-0/25	0.0	14.622500	0.0	0.0
49	WC DPT-2640-21-0.1/0.1	-0.100000	0.116980	0.0	0.0
50	WC DPT-2640-22-0.25/0.25	-0.250000	0.292450	0.0	0.0
51	WC DPT-2640-23-0.5/0.5	-0.500000	0.584900	0.0	0.0
52	WC DPT-2640-24-1/1	-1.000000	1.169800	0.0	0.0
53	WC DPT-2640-25-2.5/2.5	-2.500000	2.924500	0.0	0.0
54	WC DPT-2640-26-5/5	-5.000000	5.849000	0.0	0.0
55	WC DPT-2640-27-10/10	-10.000000	11.698000	0.0	0.0
56	WC DPT-2640-28-25/25	-25.000000	29.245000	0.0	0.0
57	WC DPT-2641-1-0/0.1	-0.025000	0.03663	0.0	0.0
58	WC DPT-2641-2-0/0.25	-0.062500	0.091575	0.0	0.0
59	WC DPT-2641-3-0/0.5	-0.125000	0.18315	0.0	0.0
60	WC DPT-2641-4-0/1	-0.250000	0.3663	0.0	0.0
61	WC DPT-2641-5-0/2.5	-0.625000	0.915750	0.0	0.0
62	WC DPT-2641-6-0/5	-1.250000	1.831500	0.0	0.0
63	WC DPT-2641-7- 0/10	-2.500000	3.663000	0.0	0.0
64	WC DPT-2641-8-0/25	-6.250000	9.157500	0.0	0.0
65	WC DPT-2641-21-0.1/0.1	-0.150000	0.073260	0.0	0.0
66	WC DPT-2641-22-0.25/0.25	-0.375000	0.183150	0.0	0.0
68	WC DPT-2641-23-0.50/0.50	-0.750000	0.366300	0.0	0.0
67	WC DPT-2641-24-1.0/1.0	-1.500000	0.732600	0.0	0.0
69	WC DPT-2641-25-2.5/2.5	-3.750000	1.831500	0.0	0.0
70	WC DPT-2641-26-5.0/5.0	-7.500000	3.663000	0.0	0.0
71	WC DPT-2641-27-10.0/10.0	-15.000000	7.326000	0.0	0.0
72	WC DPT-2641-28-25.0/25.0	-37.500000	18.315000	0.0	0.0
73	mBAR IDP250 0/62.28	-15.56875	22.76540	0.0	0.0
74	mBAR IDPB001-0.249/0.249	-0.37365	0.18212	0.0	0.0
75	mBAR IDPB002-0.623/0.623	-0.93450	0.45549	0.0	0.0
76	mBAR IDPB005-1.245/1.245	-1.86750	0.91025	0.0	0.0
Continued on next page . . .					

Linear Parameter for Standard Ranges (Cont.)					
For FPUHW, use standard ranges 26 to 33. For DCM140HW and DCMHW, use standard ranges 1 to 25 and 34 to 111.					
Range	**Type**	**Parm 1**	**Parm 2**	**Parm 3**	**Parm 4**
77	mBAR IDPB010-2.491/2.491	-3.73650	1.82123	0.0	0.0
78	mBAR IDPB025-6.228/6.228	-9.34125	4.55308	0.0	0.0
79	mBAR IDPB050-12.45/12.45	-18.67500	9.10251	0.0	0.0
80	mBAR DPT-2640-1 0/0.249	0.0	0.14570	0.0	0.0
81	mBAR DPT-2640-2 0/0.623	0.0	0.36439	0.0	0.0
82	MBAR DPT-2640-3 0/1.245	0.0	0.72820	0.0	0.0
83	MBAR DPT-2640-4 0/2.491	0.0	1.45699	0.0	0.0
84	MBAR DPT-2640-5 0/6.228	0.0	3.64246	0.0	0.0
85	MBAR DPT-2640-6 0/12.450	0.0	7.28201	0.0	0.0
86	mBAR DPT-2640-7 0/24.910	0.0	14.56986	0.0	0.0
87	mBAR DPT-2640-8 0/62.275	0.0	36.42465	0.0	0.0
88	MBAR DPT-2640-21-0.249/0.249	-0.24910	0.29140	0.0	0.0
89	mBAR DPT-2640-22-0.623/0.623	-0.62300	0.72879	0.0	0.0
90	mBAR DPT-2640-23-1.245/1.245	-1.24500	1.45640	0.0	0.0
91	mBAR DPT-2640-24-2.491/2.491	-2.49100	2.91397	0.0	0.0
92	mBAR DPT-2640-25-6.228/6.228	-6.22750	7.28493	0.0	0.0
93	mBAR DPT-2640-26-12.45/12.45	-12.45000	14.56401	0.0	0.0
94	mBAR DPT-2640-27-24.91/24.91	-24.91000	29.13972	0.0	0.0
95	mBAR DPT-2640-28-62.275/62.275	-62.27500	72.84930	0.0	0.0
96	mBAR DPT-2641-1 0/0.249	-0.06228	0.09125	0.0	0.0
97	MBAR DPT-2641-2 0/0.623	-0.15575	0.22820	0.0	0.0
Continued on next page . . .					

Linear Parameter for Standard Ranges (Cont.)					
For FPUHW, use standard ranges 26 to 33. For DCM140HW and DCMHW, use standard ranges 1 to 25 and 34 to 111.					
Range	**Type**	**Parm 1**	**Parm 2**	**Parm 3**	**Parm 4**
98	mBAR DPT-2641-3 0/1.245	-0.31125	0.45604	0.0	0.0
99	mBAR DPT-2641-4 0/2.491	-0.62275	0.91245	0.0	0.0
100	MBAR DPT-2641-5 0/6.228	-1.55688	2.28113	0.0	0.0
101	MBAR DPT-2641-6 0/12.450	-3.11250	4.56044	0.0	0.0
102	MBAR DPT-2641-7 0/24.910	-6.22750	9.12453	0.0	0.0
103	MBAR DPT-2641-8 0/62.275	-15.56875	22.81133	0.0	0.0
104	MBAR DPT-2641-21-0.249/0.249	-0.37365	0.18249	0.0	0.0
105	mBAR DPT-2641-22-0.623/0.623	-0.93450	0.45641	0.0	0.0
106	mBAR DPT-2641-23-1.245/1.245	-1.86750	0.91209	0.0	0.0
107	mBAR DPT-2641-24-2.491/2.491	-3.73650	1.82491	0.0	0.0
108	mBAR DPT-2641-25-6.228/6.228	-9.34125	4.56227	0.0	0.0
109	mBAR DPT-2641-26-12.45/12.45	-18.67500	9.12087	0.0	0.0
110	mBAR DPT-2641-27-24.91/24.91	-37.36500	18.24907	0.0	0.0
111	mBAR DPT-2641-28-62.275/62.275	-93.41250	45.62267	0.0	0.0

NC File
AOD Keyword

Analog Output Digital Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
AOD							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
Choose one:	DCM140HW						a
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	b
		CO	subslot number	integer	1-2	[1]	c
	DCMHW						a
		O	slot number	integer	1-10	[1]	
	D140IHW						a
		O	slot number	integer	1-10	[1]	
		O	step ratio	fp		[0.9]	
		O	saturation size	integer	0-255	[90]	
	DCMIHW						a
		O	slot number	integer	1-10	[1]	
		O	step ratio	fp		[0.9]	
		O	saturation size	integer	0-255	[90]	
INIT	O						
		O	auto restore	Boolean	Y/N	[Y]	
UNITS	O						
		O	engineering units	char (6)		[PCT]	
		O	decimal display posit.	integer	0-3	[1]	

Notes:
- a Use DCM140HW or DCMHW for proportional control.
  Use D140IHW or DCMIHW for incremental control.
- b 0 = AOD, 1 = multiple AOD (MAOD)
- c Subslot ignored unless multiple AOD.

Continued on next page . . .

Analog Output Digital Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
SPANS	O						
		R	low input span equat.	fp			d
		R	high input span equat.	fp			d
		R	low output span equat.	fp			d
		R	high output span equat.	fp			d
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	e
		O	comm. disable	Boolean	Y/N	[N]	
		O	override report type	integer	0-6	[0]	f

Notes:
- d The default parameter for this field is undefined, which equates to an FFFFFFFF in the database.
- e Ignored if the point history flag is No.
- f Valid report types (0-6) are:
  - 0 = no report
  - 1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
  - 5 = followup
  - 6 = status

Example

```
AOD “AC4”, “HUM”
   HARDWARE “NC1-HW”, “DCM1”
   DCMHW 3
   SPANS 0.0,50.0,0.0,100.0
```

Semantic Rules

AOD — The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist and must be a DCM on the same NCM.

SPANS — Low input < High input

Low output <> High output

DCM140HW

If point type is AOD, the slot number must be free. If point type is multiple AOD, the slot and subslot must be free.

D140IHW, DCMHW, and DCMIHW
The slot number must be free.

NC File
AOS Keyword

Analog Output Setpoint Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
AOS							
		R	system name	name			
		R	object name	name			
		O	expanded id	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
Continued on next page . . .							

Analog Output Setpoint Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (options continued on next page):	DCM140HW						
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	a
		CO	subslot number	integer	1-2	[1]	b
	DCMHW						
		O	slot number	integer	1-10	[1]	
	D140IHW						
		O	slot number	integer	1-10	[1]	
		O	step ratio	fp		[0.9]	
		O	saturation size	integer	0-255	[90]	
	DCMIHW						
		O	slot number	integer	1-10	[1]	
		O	step ratio	fp		[0.9]	
		O	saturation size	integer	0-255	[90]	
	DCDRHW						
		R	HW device type	char (8)			c
		R	HW reference	char (7)			c
		O	local control	Boolean	Y/N	[N]	d
	DSCHW						
		O	logical point type	char (3)		[INC]	e
		O	logical point number	integer	1-255	[1]	f
	FPUHW						
		O	slot number	integer	1-16	[1]	

Notes:

a 0 = AOS, 1 = multiple AOS (MAOS)

b Subslot ignored unless point type is multiple AOS.

c Valid DCDR **device types** and **hardware references** are:

LCP	OUT1-8, ACO1-4
DX9100	OUT1-14, ACO1-8, XTnAO1-8 (n = 1 - 8)
DX91ECH	OUT1-14, ACO1-8, XTnAO1-8 (n = 1 - 8)
DC9100	OUT1-14, ACO1-8
DR9100	none
TC9100	none
XT9100	AO1-8
XTM	AO1-8

d Local control is not allowed on XT9100 and XTM hardware.

e Valid logical point types for DSCHW are: ADP, INC, ASP.

f For the ASP logical point type, the only valid logical point numbers are 4 through 67.

Continued on next page . . .

Analog Output Setpoint Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (options continued from previous page):	LONHW						
		R	HW device type	char (8)			m
		R	HW reference	char (7)			m
	N2OPENHW						
		O	point type	char (3)	AO,ADF, ADI	[AO]	
		O	point address	integer	1-256	[1]	l
		O	local control	Boolean	Y/N	[N]	
INIT	O						
		O	initial value	fp		[55.0]	
		O	auto restore	Boolean	Y/N	[Y]	
		O	initial value used	Boolean	Y/N	[N]	
UNITS	O						
		O	engineering units	char (6)		[DEGF]	
		O	decimal display posit.	integer	0-3	[1]	
SPANS	O						g
		R	low input span equat.	fp			h
		R	high input span equat.	fp			h
		R	low output span equat.	fp			h
		R	high output span equat.	fp			h
FEEDBACK	O						i
		R	feedback system name	name		[null]	

Notes:

g If DCDRHW, LONHW, or N2OPENHW, do not use SPANS subkeyword.

h Default parameter for this field is undefined. Equates to an FFFFFFFF in the database.

i If FPUHW, the FEEDBACK subkeyword is required. If DCDRHW and local control = Yes, feedback assignment is not supported.

l Refer to the *Analog Output Setpoint (AOS) Object Technical Bulletin (LIT-636084)* for mapping restrictions.

m Valid LONHW device types and hardware references are LONTCU xxAOxxx, xxAPxxx, and xxAUxxx to xxAZxxx (the xs are the placeholders for specific addressing numbers and characters; see the device's technical bulletin for details). For other LONWORKS devices, see *LONWORKS Compatible Devices Supported by the NCM350 Technical Bulletin (LIT-1162100)*.

Continued on next page . . .

Analog Output Setpoint Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		R	feedback object	name		[null]	
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. hist. flag	Boolean	Y/N	[N]	j
		O	comm. disable	Boolean	Y/N	[N]	
		O	override report type	integer	0-6	[0]	k

Notes:
j Ignored if the point history flag is No.
k Valid report types (0-6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical 2, Critical 3, or Critical 4
5 = followup
6 = status

Example

```
AOS “AC3”, “HUMSET”
   HARDWARE “NC1-HW”, “DCM1”
   DCMHW 4
   INIT 50,N
```

Semantic Rules

AOS The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist, match the type of selected controller, (for example, DCM, DCM140, DSC, FPU, N2OPEN), and be on the same NCM.

SPANS Low input < High input
Low output <> High output

FEEDBACK The system\object name must already exist (WARNING). The system\object name must be an AI, AD, or ACM.

DCDRHW The hardware reference must be valid for the specified hardware device type. The hardware reference must be free. The hardware reference must be valid for an AOS.

DCM140HW

If point type is AOS, the slot number must be free. If point type is multiple AOS, the slot and subslot must be free.

D140IHW, DCMHW, and DCMIHW

The slot number must be free.

DSCHW LPN (logical point number) must be free for the logical point type chosen.

FPUHW The slot number must be free.

LONHW The hardware reference must be valid for the specified hardware device type.

The hardware reference must be free.

The hardware reference must be valid for an AOS.

N2OPENHW

Point type and address must be free.

If local control = y, FEEDBACK keyword is allowed for “Change Default” attributes (see HVAC PRO.prn output file).

IMPORTANT:	This information applies to objects mapped to DCDRHW, LONHW, and N2OPENHW. Though the following is not checked by the DDL compiler, it is important for you to consider. If you are mapping a CS object attribute and a standard object to the same hardware reference (for N2OPENHW, the hardware reference is the combination of the point type and point address), make sure the Override and Adjust flags are set to No (False) for the CS object attribute. This is to ensure that there is only one command path to the hardware reference.

Binary Data Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
BD							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ASSOCINP	O						
		R	associated object type	char (8)		[null]	a
		R	associated system name	name		[null]	
		R	associated object name	name		[null]	
		R	associated attribute	char (8)		[null]	
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
UNITS	O						
		O	engineering units S0	char (6)		[OFF]	
		O	engineering units S1	char (6)		[ON]	
INIT	O						
		O	latching point	Boolean	Y/N	[N]	
		O	initial value closed	Boolean	Y/N	[N]	
		O	normal state	integer	0-2	[0]	b

Notes:

a Valid object types are:

ACM	BI	CS	DSC8500	LCG	N2OPEN
AD	BO	DCDR	D600	LON	PIDL
AI	C210A	DCM	FIRE	MC	READER
AOD	C260A	DCM140	FPU	MSD	XM
AOS	C260X	DLLR	JCB	MSI	ZONE
BD	C500X	DSC-1000	LCD	MSO	

b Valid normal states are:
0 = none
1 = state 0
2 = state 1

Continued on next page . . .

Binary Data Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		O	alarm delay time	integer	0-255	[30]	
		O	delay all alarms	Boolean	Y/N	[N]	
		O	adjust disabled	Boolean	Y/N	[N]	
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[N]	
		CO	save pt. hist. flag	Boolean	Y/N	[N]	c
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	0-255	[0]	d
		O	normal report type	integer	0-6	[0]	e
		O	alarm report type	integer	0-6	[0]	e
		O	override report type	integer	0-6	[0]	e
LOAD	O						
		R	load group system name	name		[null]	
		R	load group object name	name		[null]	
		O	priority	integer	1-4	[3]	
		O	load locked	Boolean	Y/N	[N]	
		R	load rating	long	1-999999		
		R	min. release time (min)	integer	1-255		
		R	max. shed time (min)	integer	1-255		
COMFORT	O						
		R	comfort system name	name		[null]	
		R	comfort object name	name		[null]	
		R	min. shed time (mins.)	integer	1-255	[0]	

Notes:
- c Ignored if the point history flag is No.
- d Ignored unless the corresponding report type is not 0.
- e Valid report types (0 - 6) are:
  0 = no report
  1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
  5 = followup
  6 = status

Example

```
BD “AC4”, “RF-STAT”
ASSOCINP “BI”,“AC3”,“RF-STAT”,“VALUE”
UNITS “OFF”, “ON”
INIT N,N,1
```

Semantic Rules

BD	The system name must already exist. The object name must be unique within the system. Although not checked by the compiler, if the BD is going to map to an ALLEN-BRADLEY host system, the system name must be a number from 0 to 65535, and the object name must be a number from 800 to 3999. Although not checked by the compiler, if the BD is going to map to a JC/85/40 point, there are specific requirements for system and object name. See the *JC/85 Gateway Application Note (LIT-6363147)* for details.
ASSOCINP	The system\object must already exist (WARNING). The attribute must be a legal Boolean attribute for that object.
INIT	For a latching point, the normal state cannot be None.
LOAD	The load group reference must exist as a Demand Limiting Load Rolling (DLLR) load group object.
COMFORT	The COMFORT subkey can be used only if the LOAD subkey is also used. The referenced comfort object must exist as a defined BI, BD, AI, or AD object. The minimum shed time must be less than the maximum shed time.

NC File
BI Keyword

Binary Input Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
BI							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
Choose one (choices continued on next page):	D600HW						
		R	reader number	integer	1-16		
		R	binary point number	integer	1-8		
		O	input type	integer	2	[2]	a
		O	point enable	Boolean	Y/N	[Y]	b
		O	suppress time zone #	integer	0-8	[0]	c
		O	alarm if relay set	Boolean	Y/N	[N]	
		O	quiet if relay set	Boolean	Y/N	[N]	
	DCDRHW						
		R	HW device type	char (8)			d
		R	HW reference	char (7)			d

Notes:

a Input alarm type = 2 if 2-state alarm.

b Point enable must be Y (Yes) if N2 device is to recognize the alarm point as active. Setting point enable = N (No) prevents the binary point from operating at the N2.

c Suppress binary alarm time zone range: 0-8. Default 0 means no time zone applies.

d Valid DCDR device types and hardware references are:

LCP/DC9100	DI1-8, LCM1-4
DX9100	DI1-8, XTnDI1-8 (n = 1-8), LRS1-32
DX91ECH	DI1-8, XTnDI1-8 (n = 1-8), LRS1-64
DR9100	WIN, OCC, AIRQ
TC9100	WIN, OCC, AIRQ, ALM, AFM
XT9100	1DI1-8, 2DI1-8
XTM	1DI1-8, 2DI1-8

Continued on next page . . .

Binary Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (choices continued from previous page and continued on next page):	DCM140HW						
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	e
		CO	subslot number	integer	1-2	[1]	f
		O	debounce filter	integer	1-255	[2]	
		CO	LED On when closed	Boolean	Y/N	[Y]	g
	DCMHW						
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	e
		CO	subslot number	integer	1-2	[1]	f
		O	debounce filter	integer	1-255	[2]	
		CO	LED On when closed	Boolean	Y/N	[Y]	g
	DSCHW						
		O	logical point type	char(3)		[CON]	h
		O	logical point number	integer	1-255	[1]	
	FPUHW						
		O	slot number	integer	1-16	[1]	
		O	binary type	char (6)		[BIN101]	i
	LONHW						
		R	HW device type	char (8)			s
		R	HW reference	char (7)			s

Notes:

e Valid point types for the DCM140HW and DCMHW are:
0 = BI point
1 = Multi BI (Always define an FM-IBN-101 as a Multi-BI.)

f Subslot ignored unless the point type is Multi-BI (1).

g Ignored unless the point type is BI point (0).

h Valid entries for logical point type are: BSP, BDP, CON, and MAN.

i Valid entries for binary type are: BIN101, BIF101, BIS101, SST101, and SST102.

s Valid LONHW device types and hardware references are:
LONTCU xxBIxxx (The xs are placeholders for specific addressing numbers and characters.) See the device's technical bulletin for details. For other LONWORKS devices, see *LONWORKS Compatible Devices Supported by NCM350 Technical Bulletin (LIT-1162100)*.

Continued on next page . . .

Binary Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (choices continued from previous page):	N2OPENHW						
		O	point type	char (3)	BI	[BI]	
		O	point address	integer		[1]	r
		O	operating instruction	integer	0-32767	[0]	
	XMHW						
		R	XM type	integer	1-3		j
		O	point type	integer	1-2	[1]	k
		O	slot number	integer	1-32	[1]	
		CO	debounce filter	integer	12-3060	[24]	l
		O	LED On when closed	Boolean	Y/N	[Y]	
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
UNITS	O						
		O	status units S0	char (6)		[OFF]	
		O	status units S1	char (6)		[ON]	
INIT	O						
		O	latching point	Boolean	[Y/N]	[N]	
		O	normal state	integer	0-2	[0]	m
		O	alarm delay time	integer	0-255	[30]	
		O	delay all alarms	Boolean		[N]	

Notes:
- j Valid XM types for the XMHW are: 1 = XBN, 2 = XRM, 3 = XRL or XRE.
- k Valid point types for XMHW are: 1 = single, 2 = Form C.
- l Use the debounce filter only if the hardware type is single (1).
- m Valid normal states are:
  - 0 = none
  - 1 = state 0
  - 2 = state 1
- r Refer to *Binary Input (BI) Object Technical Bulletin (LIT-636088)* for mapping restrictions.

Continued on next page . . .

Binary Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	n
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	0-255	[0]	o
		O	normal report type	integer	0-6	[0]	p
		O	alarm report type	integer	0-6	[0]	p
		O	override report type	integer	0-6	[0]	pq

Notes:
- n Ignored if the point history flag is No.
- o Ignored if the corresponding report type is 0.
- p Valid report types (0-6) are:
  - 0 = no report
  - 1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
  - 5 = followup
  - 6 = status
- q If DCDRHW or LONHW, override report type is ignored.

Example

```
BI “LOBBY”, “DOOR”
HARDWARE “NC1-HW”,“DCM1”
DCMHW 5
UNITS “OPE”, “CLO”
INIT N,1,15
```

Semantic Rules

BI	The system name must already exist. The object name must be unique.
HARDWARE	The system\object must already exist, match the selected hardware type (for example, XMHW, N2OPENHW), and be on the same NCM as the BI.
D600HW	The binary point number must be free on the associated reader. The reader number must reference an existing card reader. If non-zero, the suppress time zone # must reference an existing time zone.
DCDRHW	The hardware reference must be valid for the specified hardware device type. The hardware reference must be free. The hardware reference must be valid for a BI.
DCM140HW	The slot number must be free. For multi BI, the slot and subslot pair must be free.
DCMHW	The slot number must be free. For multi BI, the slot and subslot pair must be free.
DSCHW	LPN must be free for the logical point type chosen.
FPUHW	The slot number must be free. If SST101or SST102, then the input side of the slot number must be free. If BIF101, BIS101, or SST102, then the INIT keyword, normal state must be 1. If BIF101, then for INIT keyword, latching point must be Y (Yes).
LONHW	The hardware reference must be valid for the specified hardware device type. The hardware reference must be free. The hardware reference must be valid for a BI.

N2OPENHW	Point type and address must be free.
XMHW	If the XM type is 2 or 3, then you must use Slots 1-8. If point type is single, the slot number must be free. If point type is Form C, the slot number must be an odd number; and the slot number and the slot number +1 must be free. The debounce filter must be divisible by 12.
INIT	For a latching point, the normal state cannot be 0.

IMPORTANT:	This information applies to objects mapped to N2OPENHW. Though the following is not checked by the DDL compiler, it is important for you to consider. If you are mapping a CS object attribute and a standard object to the same hardware reference (the hardware reference is the combination of the point type and point address), make sure the Override and Adjust flags are set to No (False) for the CS object attribute. This is to ensure that there is only one command path to the hardware reference.

NC File
BO Keyword

Binary Output Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
BO							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
Choose one (choices continued on next page):	DCM140HW						
		O	point type	integer	0-1	[0]	a
		O	slot number	integer	1-10	[1]	
		CO	pulse duration	integer	20-5100	[200]	b
		O	LED On when closed	Boolean	Y/N	[Y]	
	DCMHW						
		O	point type	integer	0-1	[0]	a
		O	slot number	integer	1-10	[1]	
		CO	pulse duration	integer	20-5100	[200]	b
		O	LED On when closed	Boolean	Y/N	[Y]	

Notes: a Valid point types for DCM140HW and DCMHW are:
0 = maintained
1 = latched

b Ignored if the point type is maintained.

Continued on next page . . .

Binary Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (choices continued from previous page):	DCDRHW						
		R	HW device type	char (8)			c
		R	HW reference	char (7)			c
		O	local control	Boolean	Y/N	[N]	d
	DSCHW						
		O	logical point type	char (3)		[MAN]	e
		O	logical point number	integer	1-255	[1]	f
	FPUHW						
		O	slot number	integer	1-16	[1]	
		O	binary type	char (6)		[SST-101]	g
	LONHW						
		R	HW device type	char (8)			o
		R	HW reference	char (7)			o
	N2OPENHW						
		O	point type	char (3)	BO,BD	[BO]	
		O	point address	Integer		[1]	n
		O	local control	Boolean	Y/N	[n]	

Notes:

c Valid DCDR **device types** and **hardware references** are:

LCP	DO3-8, STUP, SOFF, DCO1-4
DX9100	DO3-8, STUP, SOFF, DCO1-32, XTnDO1-8 (n = 1-8)
DX91ECH	DO3-8, STUP, SOFF, DCO1-32, XTnDO1-8 (n = 1-8)
DC9100	DO3-8, STUP, SOFF, DCO1-4
DR9100	DO3-7, STUP, SOFF
TC9100	DO1-7, STUP, SOFF
XT9100	1DO1-8 and 2DO1-8
XTM	1DO1-8 and 2DO1-8

d Ignored if HW device type is XTM or XT9100.

e Valid entries for DSC logical point type are: BDP, BOF, BSP, MAN, and MOM.

f If BSP logical point type, LPN must be 5, 6, or 7.

g Valid entries for FPU binary type are: SST101 and SST102.

n Refer to *Binary Output (BO) Object Technical Bulletin (LIT-636090)* for mapping restrictions.

o Valid LONHW device types and hardware references are:
LONTCU xxBOxxx, xxBPxxx, and xxBUxxx to xxBZxx (the xs are placeholders for specific addressing numbers and characters). See the device's technical bulletin for details. For other LONWORKS devices, see *LONWORKS Compatible Devices Supported by NCM350 Technical Bulletin (LIT-1162100)*.

Continued on next page . . .

Binary Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
	XMHW						
		R	XM type	integer	2-3		h
		O	slot number	integer	1-8	[1]	
		O	pulse duration	integer	12-3060	[252]	
		O	LED On when closed	Boolean	Y/N	[Y]	
UNITS	O						
		O	status units Stop 0	char (6)		[OFF]	
		O	status units Start 1	char (6)		[ON]	
		O	initial value closed	Boolean	Y/N	[N]	
		O	output relay closed for start	Boolean	Y/N	[Y]	
		O	initial value used	Boolean	Y/N	[N]	
RESET	O						
		O	auto restore flag	Boolean	Y/N	[Y]	
TIMER	O						
		O	heavy equipment delay (sec)	integer	0-255	[5]	i
		O	minimum On time (sec)	integer	0-255	[1]	i
		O	minimum Off time (sec)	integer	0-255	[0]	i
		O	max. starts per hour	integer	0-255	[255]	i j
FEEDBACK	O						
		R	feedback system name	name		[null]	
		R	feedback object name	name		[null]	
		O	feedback closed for start	Boolean	Y/N	[Y]	

Notes: h Valid XM types for XMHW are: 2 = XRM, 3 = XRL or XRE.
i Ignored if N2OPENHW or DCDRHW local control = Y.
j 0 means unlimited starts per hour.

Continued on next page . . .

Binary Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	k
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	0-255	[0]	l
		O	normal report type	integer	0-6	[0]	m
		O	alarm report type	integer	0-6	[0]	m
		O	override report type	integer	0-6	[0]	m
LOAD	O						
		R	load group system name	name		[null]	
		R	load group object name	name		[null]	
		O	priority	integer	1-4	[3]	
		O	load locked	Boolean	Y/N	[N]	
		R	load rating	long	1 - 999999		
		R	min. release time (min)	integer	1-255		
		R	max. shed time (min)	integer	1-255		
COMFORT	O						
		R	comfort system name	name		[null]	
		R	comfort object name	name		[null]	
		R	min. shed time (min)	integer	1-255	[0]	

Notes:
- k Ignored if the point history flag is No.
- l Ignored if the corresponding report type is 0.
- m Valid report types (0-6) are:
  - 0 = no report
  - 1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
  - 5 = followup
  - 6 = status

Example

```
BO “LOBBY”, “BOLT”
HARDWARE “NC1-HW”, “XRL4”
XMHW 2,8,N
UNITS “UNLOCK”, “LOCK”,N
RESET N
TIMER 0,0,0,255
```

Semantic Rules

BO	The system name must already exist. The object name must be unique.
HARDWARE	The system\object must already exist, match the selected hardware type (for example, DCMHW, FPUHW) and be on the same NCM. If an XMHW, it must be a type 2 or 3.
DCDRHW	The hardware reference must be valid for the specified hardware device type. The hardware reference must be free. The hardware reference must be valid for a BO. A maximum of 128 software objects may be mapped to one System 9100 device (for example, DX9100).
DCM140HW	The slot number must be free. The pulse duration must be divisible by 20.
DCMHW	The slot number must be free. The pulse duration must be divisible by 20.
DSCHW	LPN must be free for the logical point type chosen.
FPUHW	Both the slot number and the output side of the slot number must be free.
LONHW	The hardware reference must be valid for the specified hardware device type. The hardware reference must be free. The hardware reference must be valid for a BO.

N2OPENHW

Point type and address must be free.

XMHW The slot number must be free. The pulse number must be divisible by 12.

IMPORTANT:	This information applies to objects mapped to DCDRHW, LONHW, and N2OPENHW. Though the following is not checked by the DDL compiler, it is important for you to consider. If you are mapping a CS object attribute and a standard object to the same hardware reference (for N2OPENHW, the hardware reference is the combination of the point type and point address), make sure the Override and Adjust flags are set to No (False) for the CS object attribute. This is to ensure that there is only one command path to the hardware reference.

FEEDBACK The system\object must already exist (WARNING).

The software object must be a BI or BD.

For proper execution, if the feedback object is a BI, this BI should not be used by any other BO as a feedback object. This is not checked by DDL, GPL, or online generation.

LOAD The load group reference must already exist (WARNING).

The load group reference must be a DLLR load group object.

The minimum release time must be greater than or equal to the minimum on time (minimum on time is a parameter of the TIMER subkey).

COMFORT The COMFORT subkey can be used only if the LOAD subkey is also used.

The referenced comfort object must already exist (WARNING).

The referenced object must be a BI, BD, AI, or AD.

The minimum shed time must be greater than or equal to the minimum off time (minimum off time is a parameter of the TIMER subkey).

The minimum shed time must be less than the maximum shed time.

NC File
C210A Keyword

C210A Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
C210A							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
DISPLAY	O						
		O	default display AI	integer	1-6	[1]	
		O	default display unit	char (6)		[DEGF]	
		O	default display SP	integer	1-11	[7]	
C210AAPP	O						
		O	damper type	integer	0-2	[0]	a
		O	heat type	integer	0-1	[0]	b

Note: a Valid damper types are:
0 = no damper
1 = pressure dependent
2 = pressure independent

b Valid heat types are:
0 = no heat
1 = heat

Continued on next page . . .

C210A Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		O	fan type	integer	0-2	[0]	c
		O	setpoint type	integer	0-1	[0]	d
		O	aux. binary	Boolean	Y/N	[N]	
		O	aux. temp. sensor used	Boolean	Y/N	[N]	
		O	aux. differential pressure sensor used	Boolean	Y/N	[N]	
		O	aux. humidity sensor used	Boolean	Y/N	[N]	
		O	aux. input 0-5 VDC used	Boolean	Y/N	[N]	
		O	occupied/ unoccupied	integer	0-3	[0]	e
		O	warmup	integer	0-2	[0]	f
		O	shutdown	integer	0-4	[0]	g
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	override report type	integer	0-6	[0]	h

Notes:

c Valid fan types are:
0 = none
1 = series
2 = parallel

d Valid setpoint types are:
0 = local
1 = remote

e Valid occupied/unoccupied controls are:
0 = none
1 = local
2 = L2 command
3 = both

f Valid warmup controls are:
0 = none
1 = L2 command
2 = SAT and L2

g Valid shutdown controls are:
0 = none
1 = L2 command Open
2 = L2 command Close and local
3 = local
4 = L2 command Close

h Valid report types (0-6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
5 = followup
6 = status

Example

```
C210A “GND-FLR”, “SYS1”
HARDWARE “NC1-HW”, “DSC1”
DISPLAY 2, “DEGC”,3
C210AAPP 1,1,1,0,Y,Y,Y,N,N,1,0,1
```

Semantic Rules

C210A The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist and must be a C210A type DSC-1000. You can have no more than one C210A object per C210A type DSC-1000.

NC File
C260A Keyword

C260A Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
C260A							
		R	system name	name			
		R	object name	name			
		O	expanded id	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
DISPLAY	O						
		O	default display AI	integer	1-6	[1]	
		O	default display unit	char (6)		[DEGF]	
		O	default display SP	integer	1-8	[6]	
C260AAPP	O						
		O	fanon/compoff	integer	0-3	[0]	a
Note: a Valid fan types are: 0 = none 1 = local 2 = L2 command 3 = both							
Continued on next page . . .							

C260A Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		O	setpoint type	integer	0-1	[0]	b
		O	aux. binary latch used	Boolean	Y/N	[N]	
		O	aux. temp. sensor used	Boolean	Y/N	[N]	
		O	aux. temp. sensor 2 used	Boolean	Y/N	[N]	
		O	aux. humidity sensor used	Boolean	Y/N	[N]	
		O	aux. humidity sensor 2 used	Boolean	Y/N	[N]	
		O	aux. input 0-5 VDC used	Boolean	Y/N	[N]	
		O	occupied/ unoccupied	integer	0-3	[0]	c
		O	shutdown	integer	0-1	[0]	d
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	override report type	integer	0-6	[0]	e

Notes:

b Valid setpoint types are:
0 = local
1 = remote

c Valid occupied/unoccupied controls are:
0 = none
1 = local
2 = L2 command
3 = both

d Valid shutdown controls are:
0 = none
1 = L2 command

e Valid report types (0 - 6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
5 = followup
6 = status

Example

```
C260A “GND-FLR”, “SYS2”
HARDWARE “NC1-HW”, “DSC2”
DISPLAY 2, “DEGC”,3
C260AAPP 0,1,N,N,N,Y,Y,N,1,1
```

Semantic Rules

C260A The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist and must be a C260A type DSC-1000.

You can have no more than one C260A object per C260A type DSC.

CAUTION: C260X and C500X are also valid main keywords; employing the "X" version causes DDL to expect the parameters and subkeywords associated with the international models for which the objects are named.

NC File
CARD Keyword

Card Feature (Card) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
CARD							
		R	card ID	long	1-65535		
		O	issue level	integer	0-7	[0]	
		O	executive privilege	Boolean	Y/N	[N]	
		O	active-card time zone #	integer	0-8	[0]	a
		O	in_out state	integer	0-2	[0]	
		O	card process group	integer	0-64	[0]	b
		O	last name	char (10)		[null]	
		O	first name	char (8)		[null]	
READERS	R	R	valid readers	Boolean [16]			c
HARDWARE	R						
		R	D600 system name	name			
		R	D600 object name	name			

Notes:
- a If defaulted [0] and D600 time zones are checked (parameter set in the D600 database generation), this card is never active.
- b If card process group = 0, no JC-BASIC interlock process can be called. If not 0, the process associated with that group is called.
- c If [ ], no readers are valid for this card.

Example

```
CARD 9992,2,Y,4,1,12,“COLERIDGE”, “SAMUEL”
READERS [1,3,5,7,9,11,13,15]
HARDWARE “SECURITY”,“AC5”
```

Semantic Rules

CARD — The card ID must be unique to the Access controller hardware.

HARDWARE

- The system/object name must be an existing object.
- The object type must be Access controller (D600).

READERS — The referenced card readers must exist on the D600 specified by the HARDWARE subkeyword.

NC File
CS Keyword

Control System Object (CS) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
CS							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
MODEL	R	R	software model to use	char (8)			
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	override report type	integer	0-6	[0]	
DISPLAY	R						
		R	object display attribute	char (8)			
		R	NT command attribute	char (8)			
FORMAT	O	O	decimal display position	integer	0-3	[1]	

Example

```
CS “AHU1”,“AHU-1”,“CAFETERIA AHU”
MODEL “AHU-1”
HARDWARE “NC1N2”, “AHU-1”
GRAPHICS 0,0
REPORT N,N
DISPLAY “AI_3”, “SP_1”
FORMAT 1
```

Semantic Rules

CS — The system name must already exist. The object name must be unique.

MODEL — The software model must already exist.

The hardware type in the referenced software model must match the type of the hardware device object specified with the HARDWARE subkeyword (that is, if the hardware type in the model is AHU, the hardware device object must be AHU).

HARDWARE

The N2OPEN, LON, or DCDR system\object must already exist.

Up to 16 CS objects can be defined for each N2OPEN, LON, or DCDR object.

DISPLAY — The attributes specified must exist in the referenced software model. For example, attribute AI_4 is invalid if there are only 2 AI points in the model.

NC File
D600 Keyword

Access Controller (D600) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
D600							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	hardware address	integer	0-255		
		O	N2 trunk number	integer	1-2	[1]	
		O	poll priority	integer	0-3	[1]	a
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	stop valid card reports	Boolean	Y/N	[N]	
ACCF	O						
		O	ID encoding formula	integer	0-7	[0]	
		O	in_out readers exist	Boolean	Y/N	[N]	
		O	5-digit PIN codes	Boolean	Y/N	[N]	
		O	AC checks time zones	Boolean	Y/N	[Y]	
ACFAC	R						
		O	wiegand/ proximity	long	0-9999999	[0]	
		O	ncrypt facility code	long	0-9999999	[0]	
		O	magnetic strip facility code	long	0-9999999	[0]	
		O	card process timer	integer	0-480	[0]	
Note: a For poll priority: 0 = highest (most often polled) 3 = lowest (least often polled)							

Example

```
D600 “SECURITY”, “ACFLOOR1”, “AC FIRST
FLOOR”
ADDRESS 30,1,1
GRAPHICS 0,0
REPORT Y,N,N
ACCF 3,Y,N,Y
ACFAC 12345
```

Semantic Rules

D600 The system name must already exist. The object name must be unique.

ADDRESS The N2 trunk number must be defined as an N2 port in the NCM port definition.

The address must not be used by another N2 device on the same trunk.

A warning occurs if you use 0 as the N2 address, because an address of 0 can cause a problem in Application Specific Controllers (ASCs) with firmware prior to A03, B03, or C03.

NC File
DCDR Keyword

DCDR Device Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DCDR							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	hardware address	integer	0-255		
		O	N2 trunk number	integer	1-2	[1]	
		O	poll priority	integer	0-3	[3]	a
		R	device type	char (8)			b
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	

Notes:
a For poll priority: 0 = highest (most often polled)
3 = lowest (least often polled)

b Valid device types are: LCP, DX9100, DX91ECH, DC9100, DR9100, TC9100, XT9100*, XTM.

* Never map the XT9100 as a DCDR device hardware object when it is connected to a DX9100 or DX9120 through the XT Extension Bus.

Example

```
DCDR “NC1N2”, “LCP10”, “LCP at address 10”
ADDRESS 10,1,1,“LCP”
GRAPHICS 0,0
REPORT N,N
```

Semantic Rules

DCDR — The system name must already exist. The object name must be unique.

ADDRESS — The N2 trunk number must be defined as an N2 port in the NCM port definition.

For a DX91ECH, the N2 trunk number must be 2, and it must be defined as an N2E port in the NCM port definition. For a DX91ECH, the NCM must be an NCM300.

The address must not be used by another N2 device on the same trunk.

A warning occurs if you use 0 as the N2 address, because an address of 0 can cause a problem in ASCs with firmware prior to A03, B03, or C03.

IMPORTANT:	For System 9100 devices, you can map only 128 software objects to each controller. A compile error occurs if this limit is exceeded.

NC File
DCM140 Keyword

Digital Control Module 140 (DCM140) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DCM140							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	hardware address	integer	0-255		
		O	N2 trunk number	integer	1-2	[1]	
		O	poll priority	integer	0-3	[1]	a
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
Note: a For poll priority: 0 = highest (most often polled) 3 = lowest (least often polled)							

Example

```
DCM140 “NC1HW”,“DCM140-1”,“DCM140-in-EN1”
   ADDRESS 1,1,1
   GRAPHICS 0,0
   REPORT N,N
DCM140 “NC1HW”,“DCM140-6”,“DCM140-in-EN6”
   ADDRESS 6,1,1
```

Semantic Rules

DCM140 The system name must already exist. The object name must be unique.

To define a DCM140, the NCM (that the @NC file is for) must be an NCM200. If it is an NCM101, define the DCM140 as a DCM (using the DCM keyword). (When defined as a DCM, the DCM140 operates with DCM functionality.)

ADDRESS The N2 trunk number must be defined as an N2 port on the NCM.

The address must not be used by another N2 device on the same trunk.

A warning occurs if you use 0 as the N2 address, because an address of 0 can cause a problem in ASCs with firmware prior to A03, B03, or C03.

NC File
DCM Keyword

Digital Control Module (DCM) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DCM							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	hardware address	integer	0-255		
		O	N2 trunk number	integer	1-2	[1]	
		O	poll priority	integer	0-3	[1]	a
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
Note: a For poll priority: 0 = highest (most often polled) 3 = lowest (least often polled)							

Example

```
DCM “NC1-HW”, “DCM1”, “DCM-in-EN-1”
   ADDRESS 1,1,1
   GRAPHICS 0,0
   REPORT N,N
DCM “NC1-HW”, “DCM6”, “Right-DCM-in-EN-2”
   ADDRESS 6,1,1
```

Semantic Rules

DCM — The system name must already exist. The object name must be unique.

ADDRESS — The N2 trunk number must be defined as an N2 port on the NCM.

The address must not be used by another N2 device on the same trunk.

A warning occurs if you use 0 as the N2 address, because an address of 0 can cause a problem in ASCs with firmware prior to A03, B03, or C03.

NC File DELCARD Keyword

Delete a Card							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DELCARD							
		R	D600 system name	name			
		R	D600 object name	name			
		R	card ID	long	1-65535		

Example

```
DELCARD “SECURITY”,“AC5”,19901
```

Semantic Rules

DELCARD — The card ID must exist on the specified D600.

Delete an Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DELETE							a
		R	system name	name			
		R	object name	name			

Note: a This keyword can delete all objects created by the NC file. In addition, this keyword can delete a JC-BASIC process if the name given is a process name. When deleting a software object occupying one or more slots in a hardware device, the slots are marked free in the archive database. You may not delete a hardware object until all of its slots are free. You may not delete a READER if any BI objects are assigned to it or if any CARDs are valid for it.

Example

```
DELETE "NC1", "DEMAND"
```

Semantic Rules

DELETE The system\object must exist. Deletion is only allowed in the NC file and only during an incremental compile.

Note: When the DDL compiler deletes an object, it does not check to make sure that it is not referenced by another object or feature.

NC File
DELSUBORDINATE
Keyword

Delete a Subordinate							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DELSUBORDINATE							
		R	subordinate input system	name		[null]	
		R	subordinate input object	name		[null]	
		R	MC system name	name		[null]	
		R	MC object name	name		[null]	

Example

```
DELSUBORDINATE "WEST","HTGSPT","PNL-4", "MCO-4"
```

Semantic Rules

DELSUBORDINATE The MC system\object must exist and must be object type MC on the NCM being compiled. The subordinate system\object must exist in the MC's subordinate list. If the subordinate exists more than once in the MC's subordinate list, all instances of the subordinate are deleted.

NC File
DELTZ Keyword

Delete a Time Zone							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DELTZ							
		R	D600 system name	name			
		R	D600 object name	name			
		R	time zone record #	integer	1-8		

Example

```
DELTZ "SECURITY","AC5",6
```

Semantic Rules

DELTZ The Time Zone must exist on the specified D600.

Do not delete a TIMEZONE if any CARD or READER is using it.

NC File
DLLR Keyword

Demand Limiting/Load Rolling (DLLR) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DLLR	O						
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
METER	O						
		R	meter system name	name		[null]	
		R	meter object name	name		[null]	
EOI	O						
		R	EOI system name	name		[null]	
		R	EOI object name	name		[null]	
		O	target elevation in %	integer	0-99	[10]	
TARGET	O						
		R	target level system name	name		[null]	
		R	target level object name	name		[null]	
Choose one or both:	LOADROLL						
		R	load rolling target	long	0-999999		a
		CO	load rolling target level 2	long	0-999999	[ffffff]	b
		CO	load rolling target level 3	long	0-999999	[ffffff]	b
		CO	load rolling target level 4	long	0-999999	[ffffff]	b
Notes: a The default is undefined (stored as ffffff). Can be defaulted only when the subkey is defaulted. b Ignored if target level object is undefined.							
Continued on next page . . .							

Demand Limiting/Load Rolling (DLLR) Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
	DEMAND						
		R	demand limiting target	long			a
		CO	demand limiting target level 2	long	0-999999	[ffffff]	b
		CO	demand limiting target level 3	long	0-999999	[ffffff]	b
		CO	demand limiting target level 4	long	0-999999	[ffffff]	b
		O	demand interval	3-60		[15]	
		O	sensitivity	char (6)	low/ medium/ high	[medium]	
		O	tracking period	1-5		[4]	c
GRAPHICS		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
	O						
REPORT		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message number	integer	0-255	[0]	d
		O	normal report type	integer	0-6	[0]	
		O	alarm report type	integer	0-6	[0]	

Notes:
- a The default is undefined (stored as ffffff). Can be defaulted only when the subkey is defaulted.
- b Ignored if target level object is undefined.
- c For tracking period: 1 = hourly, 2 = daily, 3 = weekly, 4 = monthly, 5 = manual
- d Ignored if corresponding report type is 0.

Example

```
DLLR “NC1CNTRL”, “DLLR1”, “DLLR-1”
METER “NC1-HW”, “ACM-4”
DEMAND
GRAPHICS 0,0
REPORT N,N
```

Semantic Rules

DLLR — The system name must already exist. The object name must be unique.

There can be up to four DLLR load groups defined in an NCM.

Note: Each load group can have up to 500 loads anywhere on the network. Each NCM can have up to 500 loads.

METER — The system\object name must already exist. The meter must be an ACM, AD, or AI.

The meter must be on the same NCM as the DLLR load group.

A meter must be defined if the DEMAND subkey is used (otherwise the meter is optional).

EOI — The system\object name must exist and must be a BI object.

The EOI must be on the same NCM as the DLLR load group. (The NCM must be a System 9100 NCM or an NCM200 that supports DLLR, although DDL does not make this check.)

TARGET — The system\object name must already exist. The target must be a Multistate Data (MSD) or Multistate Input (MSI).

DEMAND — If DEMAND is defined, a meter is required (otherwise a meter is optional).

C210A and C260A Hardware Object Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DSC							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	device address	integer	1-63		
		O	L2 trunk number	integer	1-2	[1]	d
		R	device type	integer	0-1		a
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	

Notes:

a Valid device types are:
0 = C210A
1 = C260A

CAUTION: The device type also allows 2 and 3, but these parameters refer to international models. Since DDL will not error out a 2 or 3, a mistake is not indicated if either of those values is entered.

d Fire panels must be defined as follows for correct operation:
For Fire NCM, type:
Trunk 1 = N2 connection for fire panels
Trunk 2 = Fire Net connection for fire panels.
Other N2 devices on Trunk 2 work normally.

Example

```
DSC “NC1-HW”,“DSC1”
ADDRESS 2,,0
DSC “NC1-HW”,“DSC2”
ADDRESS1,,1
```

Semantic Rules

DSC — The system name must already exist. The object name must be unique.

ADDRESS — The L2 trunk number must be defined as an L2 on the NCM port definition.

The address must not be used by another L2 device on the same trunk.

NC File
DSC8500 Keyword

DSC8500 Hardware Object Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DSC8500	R						
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
S2DIRECT							
		R	device address	integer	1-63		
		O	S2 trunk number	integer	1-2	[1]	
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	

Example

```
DSC8500 “JC85HW”, “DSC20”
S2DIRECT 20,1
GRAPHICS 90,172
REPORT Y,N
```

Semantic Rules

DSC8500 — The system name must already exist. The object name must be unique.

S2DIRECT The S2 trunk number must be defined as an S2 on the NCM port definition.

The address must not be used by another S2 device on the same trunk.

For a DSC8500 connected to an NCM401, the trunk number can be 1 or 2. For a DSC8500 connected to an NCM200, the trunk number must be 2.

NC File
FIRE Keyword

Fire Controller Hardware Object Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
FIRE							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	N2 address	integer	0-255		
		O	N2 trunk number	integer	1-2	[1]	
		O	poll priority	integer	0-3	[1]	a
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[Y]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	Intelligent Fire Controller (IFC) operator report enabled	Boolean	Y/N	[Y]	
		CO	enter alarm message #	integer	0-255	[0]	b
		CO	leave alarm message #	integer	0-255	[0]	b
		CO	enter trouble message #	integer	0-255	[0]	b
Notes: a For poll priority: 0 = highest (most often polled) 3 = lowest (least often polled)							
b Ignored if corresponding report type is 0.							
Continued on next page . . .							

Fire Controller Hardware Object Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		CO	leave trouble message #	integer	0-255	[0]	b
		O	alarm report type	integer	0-6	[1]	c
		O	trouble report type	integer	0-6	[2]	c
		O	alarm event report type	integer	0-6	[6]	c
		O	normal report type	integer	0-6	[4]	c
		O	trouble event report type	integer	0-6	[6]	c
		O	normal event report type	integer	0-6	[6]	c
IFC2020	O						
		O	last forward activated zone	integer	1-239	[200]	
Note: c Valid report types (0-6) are: 0 = no report 1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4 5 = followup 6 = status							

Note: When you use DDL to define ZONE objects, you must also configure the IFC-2020 to reflect the same zone definitions and the same last forward zone. You may need to add or change interlock statements. For more information, see the *IFC-1010/2020 Programming Technical Bulletin (LIT-448060)*.

Example

```
FIRE “NC1-HW”, “IFC2”
ADDRESS 10,2,1
GRAPHICS 67,10
REPORT Y,N,Y,37,38,39,40,1,2,6,4
IFC2020 200
```

Semantic Rules

FIRE — The system name must already exist. The object name must be unique.

ADDRESS — The N2 trunk number must be defined as an N2 on the NCM port definition.

The address must not be used by another N2 device on the same trunk.

A warning occurs if you use 0 as the N2 address, because an address of 0 can cause a problem in ASCs with firmware prior to A03, B03, or C03.)

IFC2020 — For semantic rules and more information on last forward activated zone, see the *IFC-1010/2020 Programming Technical Bulletin (LIT-448060)*.

Note: Poll priority 0 or 1 is recommended. Lower polling rates may result in degraded performance.

NC File
FPU Keyword

FPU Hardware Object Definition

Keyword	KWOPT	R/O/C	Parameter	Type	Restrict	Def	Note
FPU	R						
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	device address	integer	0-63		
		O	S2 trunk number	integer	1-2	[1]	
		O	S2 device type	char (4)		[FPU]	a
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	

Note: a Valid entries are FPU, FFPU, or SFPU.

Example

```
FPU "JCHW", "FPU2"
ADDRESS 10,1,"FPU"
GRAPHICS 70,200
REPORT Y,N
```

Semantic Rules

FPU — The system name must already exist. The object name must be unique.

ADDRESS — The S2 trunk number must be defined as an S2 on the NCM port definition.

The address must not be used by another S2 device on the same trunk.

For an FPU connected to a NCM401, the trunk number can be 1 or 2. For a FPU connected to an NCM200, the trunk number must be 2.

NC File
JCB Keyword

JC-BASIC Processes							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
JCB							a
		R	system name	name			
		R	object name	name			
Note: a	The JCB keyword defines a process system\object name.						

Example

```
JCB "AC3", "SYS4"
```

Semantic Rules

JCB — The system name must exist. The object name must be unique.

Lighting Control Device (LCD) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
LCD							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	device address	integer	0-89		
		O	N2 trunk number	integer	1-2	[1]	
		O	poll priority	integer	0-3	[3]	a
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
Note: a For poll priority: 0 = highest (most often polled) 3 = lowest (least often polled)							

Example

```
LCD “NC1-HW”,“LCD1”
ADDRESS 3,1,1
```

Semantic Rules

LCD — The system name must already exist. The object name must be unique.

ADDRESS — The N2 trunk number must be defined as an N2 port in the NCM.

The address must not be used by another N2 device on the same trunk.

A warning occurs if you use 0 as the N2 address, because an address of 0 can cause a problem in ASCs with firmware prior to A03, B03, or C03.)

NC File
LCG Keyword

Lighting Control Group (LCG) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
LCG							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GROUP	O						
		O	group number	integer	1-32	[1]	
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
OUTPUTS	O						
		O	controlled output	Boolean [40]		[ ]	a
		O	outputs to blink	Boolean [40]		[ ]	a
CLNGCREW	O						
		O	cleaning crew	Boolean [32]		[ ]	a
INPUT	O						
		O	associated input	integer	0-32	[0]	
		CO	associated input type	integer	1-3	[1]	b

Notes:
- a For the Boolean [N] parameter type, list the lighting control groups that correspond to Yes. For example, [1,9,15].
- b Ignored if the associated input is 0. Valid associated input types are:
  1 = maintained
  2 = momentary
  3 = double momentary

Continued on next page . . .

Lighting Control Group Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		CO	associated off input	integer	0-32	[0]	c
OVERRIDE	O						
		O	override delay	integer	0-5999	[60]	
		O	override blink	Boolean	Y/N	[N]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	normal report type	integer	0-6	[0]	d
		O	override report type	integer	0-6	[0]	d
EVENT	O						e
	O	event type	char (1)	H/R	[R]		

Notes:

c Ignored unless the associated input type is double momentary (3).

d Valid report types (0 - 6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
5 = followup
6 = status

e 0-4 EVENT subkeys are allowed for each LCG keyword.
The first subkey is Event 1, the second is Event 2, etc.

Continued on next page . . .

Lighting Control Group Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		CR	program day	Boolean (7)		[ ]	f
		R	start time hour	integer	0-23	[0]	
		R	start time minute	integer	0-59	[0]	
		R	stop time hour	integer	0-23	[0]	
		R	stop time minute	integer	0-59	[0]	
		O	program type	integer	0-3	[0]	g

Notes:

f Ignored if event type = H (Holiday). Required if event type = R (Regular). Valid program days (1-7) are:
1 = Sunday
2 = Monday
3 = Tuesday
4 = Wednesday
5 = Thursday
6 = Friday
7 = Saturday

g 0 = On, blink enabled
1 = On, blink disabled
2 = On, override input
3 = Off, override input

Example

```
LCG “ZONES”, “WALL”
HARDWARE “NC1-HW”, “LCD1”
GROUP 10
OUTPUTS [1,5,9,12,25,29],\
[1,5,9,12,25,29]
INPUT 1,1,0
CLNGCREW [9,25]
EVENT “R”,[2,7],10,5,20,30,0
```

Semantic Rules

LCG — The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist and must be an LCD on the same NCM.

GROUP — The group number must be free.

OUTPUTS — An output must be controlled in order for it to be designated to blink.

INPUT — The associated input and the associated off input cannot be the same if the associated input is greater than 0.

The associated input and the associated off input may not be the same as those of another LCG.

CLNGCREW

For any group, the cleaning crew input must not be the associated input or the associated off input of the group being defined.

Note: The Lighting Controller supports only Holiday and Regular schedules. All days designated as Alternate days are treated as Regular days by the Lighting Controller.

NC File
LON Keyword

LONWORKS Compatible Device (LON) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
LON							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	logical device address	integer	0-255		a
		R	neuron ID	char (12)			b
		O	LON trunk number	integer	2	[2]	
		O	poll priority	integer	1-3	[3]	c
		R	device type	char (8)			
		O	retry time	integer	16-3072	[96]	d
		O	retry number	integer	0-15	[3]	e
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	time synch.	Boolean	Y/N	[N]	
		O	authentic msg.	Boolean	Y/N	[N]	

Notes:

a This field is not used for standard LONWORKS devices; recommended value set to 0.

b Valid characters in the Neuron ID are digits 0-9 and A,B,C,D,E,F (hex address).

c For poll priority: 1 = highest (most often polled)
3 = lowest (least often polled)

d Valid retry times (in msec.) are:
16 24 32 48 64 96 128 192 256 384 512 768 1024 1536 2048 3072

e The maximum number of messages sent is one more than the number specified.

Example

```
LON “NC1LON”, “TCU10”, “TCU number 10”
ADDRESS 0, “123456789ABC”, 2, 1, “LONTCU”
GRAPHICS 0, 0
REPORT N, N, N, N
```

Semantic Rules

LON — The system name must already exist. The object name must be unique.

ADDRESS — If you specify a non-zero logical device address, the address must not be used by another LONWORKS compatible device on the same trunk (do not use address 0).

The LON trunk number must be defined as a LON port in the NCM port definition. The NCM must be a LONWORKS compatible NCM (NCM350/NCM361).

The device type must be a hardware model that is present in the hardware model list. The hardware model list is part of the static database. See the Static Subdirectory table in the *Operator Workstation Disk Layout for DDL* section of the *DDL Programmer’s Manual Compiler (LIT-630030)* section.

REPORT — The time synch flag specifies whether time synchronization messages should be sent to the LONWORKS compatible device. Only devices offering a specific network variable (NV) support this feature.

Before setting this flag to Y (Yes), make sure that the device supports this feature. Time synchronization must be defined in the mapping file of the specified LONWORKS compatible hardware device type.

Multiple Command Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
MC							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
		O	number of states	integer	2-4	[2]	a
ASSOCINP	O						
		R	associated object type	char (8)		[null]	b
		R	associated input system	name		[null]	
		R	associated input object	name		[null]	
		R	associated input attribute	char (8)		[null]	
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	

Notes: a Valid number of states are:
2 = 2 state
3 = 3 state
4 = 4 state

If there is an associated object, the number of states cannot be greater than the associated object's number of states. This is **not** checked by the DDL compiler (the associated object may not exist at compile time and, thus, cannot be checked, so checking has been omitted altogether).

b Valid object types are:

ACM	BO	DCM	FPU	MSI
AD	C210A	DCM140	JCB	MSO
AI	C260A	DLLR	LCD	N2OPEN
AOD	C260X	DSC-1000	LCG	PIDL
AOS	C500X	DSC8500	LON	READER
BD	CS	D600	MC	XM
BI	DCDR	FIRE	MSD	ZONE

Continued on next page . . .

Multiple Command Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
UNITS	O						
		O	engineering units S0	char (6)		[S0]	
		O	engineering units S1	char (6)		[S1]	
		O	engineering units S2	char (6)		[S2]	
		O	engineering units S3	char (6)		[S3]	
INIT	O						
		O	latching point	Boolean	Y/N	[N]	
		O	normal state	integer	0-4	[0]	c
		O	alarm delay time	integer	0-255	[30]	
		O	initial value	integer	0-3	[0]	
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	d
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	0-255	[0]	e
		O	normal report type	integer	0-6	[0]	f
		O	alarm report type	integer	0-6	[0]	f
		O	override report type	integer	0-6	[0]	f

Notes:

c Valid normal states are:
0 = none
1 = state 0
2 = state 1
3 = state 2
4 = state 3

d Ignored if the point history flag is No.

e Ignored unless the corresponding report type is not 0.

f Valid report types (0-6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
5 = followup
6 = status

Continued on next page . . .

Multiple Command Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
LOAD	O						
		R	load group system name	name		[null]	
		R	load group object name	name		[null]	
		O	priority	integer	1-4	[3]	
		O	load locked	Boolean	Y/N	[N]	
		R	load rating state 1	long	1-999999		
		O	load rating state 2	long	1-999999	[0]	
		O	load rating state 3	long	1-999999	[0]	
		O	shed state	integer	0-2	[0]	
		R	min. release time (min)	integer	1-255		
		R	max. shed time (min)	integer	1-255		
COMFORT	O						
		R	comfort system name	name		[null]	
		R	comfort object name	name		[null]	
		R	min. shed time (min)	integer	1-255	[0]	

Example

```
MC “PNL-4”,“MC-4”,“Expanded ID”,4
ASSOCINP “MSI”,“AC3”,“RF-SWCH”,“VALUE”
UNITS “OFF”,“SLOW”,“MED”,“FAST”,
INIT Y,3
```

Note: Once you have defined the MCO using the MC keyword, define the associated subordinate objects and commands using the SUBORDINATE keyword. See the *NC File Subordinate Keyword* section, later in this document, for information.

Semantic Rules

MC	The system name must already exist and be on the NCM for which the file is being compiled. The object name must be unique within the system.
ASSOCINP	The system\object must already exist (WARNING).
	If the number of states = 2, the attribute type must be binary or integer; otherwise, the attribute type must be integer.
INIT	For a latching point, the normal state cannot be none.
	The normal state must be less than or equal to the number of states.
	The initial value must be less than the number of states.
LOAD	The load group reference must already exist (WARNING).
	The load group reference must be a DLLR load group object.
	If there are multiple states, the load rating of the second state must be greater than the load rating of the first state, and the load rating of the third state must be greater than the load rating of the second state.
COMFORT	The COMFORT subkey can be used only if the LOAD subkey is also used.
	The referenced comfort object must already exist (WARNING).
	The referenced comfort object must be a BI, BD, AI, or AD.
	The minimum shed time must be less than the maximum shed time.

Note: Use SUBORDINATE keyword to add subordinates and theircommands to an MC object.

Multistate Data Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
MSD							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
		O	number of states	integer	2-4	[2]	a
ASSOCINP	O						
		R	associated object type	char (8)		[null]	b
		R	associated input system	name		[null]	
		R	associated input object	name		[null]	
		R	associated input attribute	char (8)		[null]	
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	

Notes:

a Valid number of states are:
2 = 2 state
3 = 3 state
4 = 4 state
If there is an associated object, the number of states cannot be greater than the associated object's number of states. This is **not** checked by the DDL compiler (the associated object may not exist at compile time and thus cannot be checked, so checking has been eliminated altogether).

b Valid object types are:

ACM	BO	DCM	FPU	MSI
AD	C210A	DCM140	JCB	MSO
AI	C260A	DLLR	LCD	N2OPEN
AOD	C260X	DSC-1000	LCG	PIDL
AOS	C500X	DSC8500	LON	READER
BD	DCDR	FIRE	MC	XM

Continued on next page . . .

Multistate Data Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
UNITS	O						
		O	engineering units S0	char (6)		[S0]	
		O	engineering units S1	char (6)		[S1]	
		O	engineering units S2	char (6)		[S2]	
		O	engineering units S3	char (6)		[S3]	
INIT	O						
		O	latching point	Boolean	Y/N	[N]	
		O	normal state	integer	0-4	[0]	c
		O	alarm delay time	integer	0-255	[30]	
		O	initial value	integer	0-3	[0]	
		O	delay all alarms	Boolean	Y/N	[N]	
		O	adjust disabled	Boolean	Y/N	[N]	
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	d
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	0-255	[0]	e
		O	normal report type	integer	0-6	[0]	f
		O	alarm report type	integer	0-6	[0]	f
		O	override report type	integer	0-6	[0]	f

Notes:

c Valid normal states are:
0 = none
1 = state 0
2 = state 1
3 = state 2
4 = state 3

d Ignored if the point history flag is No.

e Ignored unless the corresponding report type is not 0.

f Valid report types (0-6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
5 = followup
6 = status

Continued on next page . . .

Multistate Data Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
LOAD	O						
		R	load group system name	name		[null]	
		R	load group object name	name		[null]	
		O	priority	integer	1-4	[3]	
		O	load locked	Boolean	Y/N	[N]	
		R	load rating state 1	long	1-999999		
		O	load rating state 2	long	1-999999	[0]	
		O	load rating state 3	long	1-999999	[0]	
		O	shed state	integer	0-2	[0]	
		R	min. release time (min)	integer	1-255		
		R	max. shed time (min)	integer	1-255		
COMFORT	O						
		R	comfort system name	name		[null]	
		R	comfort object name	name		[null]	
		R	min. shed time (min)	integer	1-255	[0]	

Example

```
MSD “AC4”,“RF-SWCH”,“Expanded ID”,4
ASSOCINP “MSI”,“AC3”,“RF-SWCH”,“VALUE”
UNITS “OFF”,“SLOW”,“MED”,“FAST”,
INIT Y,3
```

Semantic Rules

MSD — The system name must already exist and be on the NCM for which the file is being compiled. The object name must be unique within the system.

ASSOCINP	The system\object must already exist (WARNING).
	If the number of states = 2, the attribute type must be binary or integer; otherwise, the attribute type must be integer.
INIT	For a latching point, the normal state cannot be none.
	The normal state must be less than or equal to the number of states.
	The initial value must be less than the number of states.
LOAD	The load group reference must already exist (WARNING).
	The load group reference must be a DLLR load group object.
	If there are multiple states, the load rating of the second state must be greater than the load rating of the first state, and the load rating of the third state must be greater than the load rating of the second state.
COMFORT	The COMFORT subkey can be used only if the LOAD subkey is also used.
	The referenced comfort object must already exist (WARNING).
	The referenced comfort object must be a BI, BD, AI, or AD.
	The minimum shed time must be less than the maximum shed time.

NC File MSI
Keyword

Multistate Input Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
MSI							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
		O	number of states	integer	2-4	[2]	a
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
Choose one (choices continued on next page):	DCDRHW						
		R	device type	char (8)			b
		R	hardware reference	char (7)			c
		O	wired 0	Boolean	Y/N	[N]	d

Notes:

a Valid number of states are: 2 = 2 state; 3 = 3 state, 4 = 4 state.

b Valid device types for DCDRHW are: LCP, DX9100, DX91ECH, DC9100, DR9100, TC9100, XT9100, and XTM.

c Valid hardware references are as follows:

LCP	DI1-8, LCM1-4
DX9100	DI1-8, XTnDI1-8 (n = 1-8), LRS1-32
DX91ECH	DI1-8, XTnDI1-8 (n = 1-8), LRS1-64
DC9100	DI1-8, LCM1-4
DR9100	WIN, OCC, AIRQ
TC9100	WIN, OCC, AIRQ, ALM, AFM
XT9100	1DI1-8 and 2DI1-8
XTM (2-states)	1DI1-8 and 2DI1-8
XTM (2 states with wired 0)	1DI1, 1DI3, 1DI5, 1DI7, 2DI1, 2DI3, 2DI5, 2DI7
XTM (3 or 4 states with or without wired 0)	1DI1, 1DI5, 2DI1, 2DI5

d Ignored if device type other than XTM.

Continued on next page . . .

Multistate Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (continued from previous page) (continued on next page):	DCM140HW						
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	e
		CO	subslot number	integer	1-2	[1]	f
		O	debounce filter	integer	1-255	[2]	
		CO	LED On when closed	Boolean	Y/N	[Y]	g
	DCMHW						
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	e
		CO	subslot number	integer	1-2	[1]	f
		O	debounce filter	integer	1-255	[2]	
		CO	LED On when closed	Boolean	Y/N	[Y]	g
	LONHW						
		R	device type	char (8)			p
		R	hardware reference	char (7)			p

Notes:

e Valid point types for the DCMHW are:

0 = BI point

1 = Multi BI (2 binary inputs connected to an IBN FM module in the FM slot)

f Use the subslot number only if the point type is Multi BI (1).

g Ignored unless the point type is BI point (0).

p Valid LONHW device types and hardware references are:

LONTCU xxMIxxx (The xs are placeholders for specific addressing numbers and characters; see the device's technical bulletin for details.) See *LONWORKS Compatible Devices Supported by NCM350 Technical Bulletin (LIT-1162100)*.

Continued on next page . . .

Multistate Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (continued from previous page):	XMHW						
		R	XM type	integer	1-7		h
		O	slot number	integer	1-32	[1]	i
		O	debounce filter	integer	12-3060	[24]	
		CO	LED On when closed	Boolean	Y/N	[Y]	j
		O	wired 0	Boolean	Y/N	[N]	i
UNITS	O						
		O	status units S0	char (6)		[S0]	
		O	status units S1	char (6)		[S1]	
		O	status units S2	char (6)		[S2]	
		O	status units S3	char (6)		[S3]	
INIT	O						
		O	latching point	Boolean	[Y/N]	[N]	
		O	normal state	integer	0-4	[0]	k
		O	alarm delay time	integer	0-255	[30]	
		O	delay all alarms	Boolean		[N]	

Notes:

h Valid XM types for the XMHW are:
1 = XBN
2 = XRM
3 = XRL or XRE
4 = XRL 2x2
5 = XRL 2x3
6 = XRM 2x2
7 = XRM 2x3

i For slot number restrictions, see table under *Semantic Rules* for XMHW keyword.

j Ignored unless number of states is 2 and wired 0 is No, because Yes (the default value) is used in all other cases.

k Valid normal states are:
0 = none
1 = state 0
2 = state 1
3 = state 2
4 = state 3

Continued on next page . . .

Multistate Input Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	l
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	0-255	[0]	m
		O	normal report type	integer	0-6	[0]	n
		O	alarm report type	integer	0-6	[0]	n
		O	override report type	integer	0-6	[0]	n, o

Notes:
- l Ignored if the point history flag is No.
- m Ignored if the corresponding report type is 0.n
- n Valid report types (0-6) are:
  0 = no report
  1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
  5 = followup
  6 = status
- o Ignored if DCDRHW or LONHW.

Example

```
MSI “LOBBY”,“DOOR”,“Expanded_ID”,4
HARDWARE “NC1-HW”,“DCM1”
GRAPHICS 0,0
XMHW 5,9,24,Y,N
UNITS “OFF”,“SLOW”,“MED”,“FAST”
INIT N,2
```

Semantic Rules

MSI — The system name must already exist and be on the NCM for which the file is being compiled. The object name must be unique.

HARDWARE

The system\object must already exist, match the selected hardware type (for example, XMHW, DCMHW), and be on the same NCM as the MSI. If the hardware type is DCMHW or DCM140HW, the number of states must be 2.

DCDRHW — Only the XTM device type supports more than two states.

The hardware reference must be a valid Item in the hardware model of the specified device type and must be valid for an MSI with the specified number of states and wired 0 (if specified). All hardware references used by the MSI must be free. If wired, 0 is used (XTM only), then an additional hardware reference (Item number) must be free: Item number of the MSI + number of states -1.

DCMHW — The slot number must be free. For multi BI, the slot and subslot pair must be free.

LONHW — The hardware reference must be valid for the specified hardware device type.

The hardware reference must be free.

The hardware reference must be valid for an MSI.

Note: DDL does not check whether the hardware reference supports the specified number of states.

XMHW The debounce filter must be divisible by 12.

See the table below to determine valid slot numbers for the XM hardware:

Number of States	XM Type	Wired 0	Slot Numbers
2	XBN	N	1-32
2	XBN	Y	1-31, odd numbered
2	not XBN	N	1-8
2	not XBN	Y	1-7, odd numbered
3 or 4	XBN	Y or N	1, 5, 9, 13, 17, 21, 25, and 29
3 or 4	not XBN	Y or N	1 and 5

For 2 states, the slot number must be free.

For 3 states, the slot number and the slot number + 1 must be free.

For 4 states, the slot number, the slot number + 1, and the slot number + 2 must be free.

If wired 0 is Yes, then an additional slot must be free (slot number + number of states -1).

INIT For a latching point, the normal state cannot be none.

The normal state must be less than or equal to the number of states.

Multistate Output Software Object							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
MSO							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
		O	number of states	integer	2-4	[2]	a
		O	local contact	Boolean	Y/N	[N]	c
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
		CR	local contact HW system	name			b
		CR	local contact HW object	name			b
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	

Notes:
a Valid number of states are:
2 = 2 state
3 = 3 state
4 = 4 state
b Ignored if local contact is No.
c Not valid for MSO on LONWORKS compatible devices.

Continued on next page . . .

Multistate Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one (Choices continued on next page):	DCDRHW						
		R	device type	char (8)			d
		R	hardware reference	char (7)			e
		O	local control	Boolean	Y/N	[N]	f
	DCM140HW						
		O	point type	integer	0-1	[0]	g
		O	slot number	integer	1-10	[1]	
		CO	pulse duration	integer	20-5100	[200]	h
		O	LED On when closed	Boolean	Y/N	[Y]	
	DCMHW						
		O	point type	integer	0-1	[0]	g
		O	slot number	integer	1-10	[1]	
		CO	pulse duration	integer	20-5100	[200]	h
		O	LED On when closed	Boolean	Y/N	[Y]	
	LONHW						
		R	device type	char (8)			v
		R	hardware reference	char (7)			v

Notes:

d Valid device types for DCDRHW (SYS91) and DCDRLC are: LCP, DX9100, DX91ECH, DC9100, DR9100, TC9100, XT9100, and XTM.

e Valid hardware references are as follows:

LCP	DO3-8, STUP, SOFF, DCO1-4
DX9100	DO3-8, STUP, SOFF, DCO1-32, XTnDO1-8 (n = 1-8)
DX91ECH	DO3-8, STUP, SOFF, DCO1-32, XTnDO1-8 (n = 1-8)
DC9100	DO3-8, STUP, SOFF, DCO1-4
DR9100	DO3-7, STUP, SOFF
TC9100	DO1-7, STUP, SOFF
XT9100	1DO1-8 and 2DO1-8
XTM (2-state)	1DO1-8 and 2DO1-8
XTM (3 or 4 state)	1DO1, 1DO5, 2DO1, 2DO5

f Ignored if device type is XTM or XT9100.

g Valid point types for DCMHW are:

0 = maintained (called latched in hardware)
1 = latched (called momentary in hardware)

h Ignored if the point type or XM type is maintained (latched in hardware).

v Valid LONHW device types and hardware references are:
LONTCU xxMOxxx, xxMPxxx, and xxMUxxx to xxMZxxx (The xs are placeholders for specific addressing numbers and characters.) See the device's technical bulletin for details. See *LONWORKS Compatible Devices Supported by NCM350 Technical Bulletin (LIT-1162100).*

Continued on next page . . .

Multistate Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
Choose one only if local contact is Yes.	XMHW						
		R	XM type	integer	2-7		i
		O	slot number	integer	1-8	[1]	
		O	pulse duration	integer	12-3060	[252]	h
		CO	LED On when closed	Boolean	Y/N	[Y]	j
	DCDRLC						
		R	device type	char (8)			d
		R	hardware reference	char (7)			k
		O	local contact wired 0	Boolean	Y/N	[N]	l
	DCM140LC						
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	m

Notes:

d Valid device types for DCDRHW (SYS91) and DCDRLC are: LCP, DX9100, DX91ECH, DC9100, DR9100, TC9100, XT9100, and XTM.

h Ignored if the point type or XM type is maintained (latched in hardware).

i Valid XM types for XMHW (2-7) and XMLC (1-7) are:
1 = XBN (for XMLC only)
2 = XRM
3 = XRL or XRE
4 = XRL 2x2
5 = XRL 2x3
6 = XRM 2x2
7 = XRM 2x3

j Ignored if number of states is other than 2.

k Valid hardware references are as follows:

LCP	DI1-8, LCM1-4
DX9100	DI1-8, LRS1-32
DX91ECH	DI1-8, XTnDI1-8 (n = 1-8), LRS1-64
DC9100	DI1-8, LCM1-4
DR9100	WIN, OCC, AIRQ
TC9100	WIN, OCC, AIRQ, ALM, AFM
XT9100	1DI1-8 and 2DI1-8
XTM	1DI1-8 and 2DI1-8
XTM (with wired 0)	1DI1, 1DI3, 1DI5, 1DI7 2DI1, 2DI3, 2DI5, 2DI7

l Ignored if DCDRLC device type is other than XTM.

m Valid point types for DCMLC are:
0 = BI point
1 = Multi BI (two binary inputs connected to an IBN FM module in the FM slot)

Continued on next page . . .

Multistate Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		CO	subslot number	integer	1-2	[1]	n
		O	debounce filter	integer	1-255	[2]	
		CO	LED On when closed	Boolean	Y/N	[Y]	o
	DCMLC						
		O	slot number	integer	1-10	[1]	
		O	point type	integer	0-1	[0]	m
		CO	subslot number	integer	1-2	[1]	n
		O	debounce filter	integer	1-255	[2]	
		CO	LED On when closed	Boolean	Y/N	[Y]	o
	XMLC						
		R	XM type	integer	1-7		i
		O	slot number	integer	1-32	[1]	
		CO	debounce filter	integer	12-3060	[24]	p
		O	LED On when closed	Boolean	Y/N	[Y]	
		O	local contact wired 0	Boolean	Y/N	[N]	
UNITS	O						
		O	status units S0	char (6)		[S0]	
		O	status units S1	char (6)		[S1]	
		O	status units S2	char (6)		[S2]	
		O	status units S3	char (6)		[S3]	
		CO	output relays closed for start	Boolean	Y/N	[Y]	j
		O	initial value	integer	0-3	[0]	
		O	initial value used	Boolean	Y/N	[N]	
RESET	O						
		O	auto restore flag	Boolean	Y/N	[Y]	

Notes: i Valid XM types for XMHW (2-7) and XMLC (1-7) are:

1 = XBN (for XMLC only)
2 = XRM
3 = XRL or XRE
4 = XRL 2x2
5 = XRL 2x3
6 = XRM 2x2
7 = XRM 2x3

j Ignored if number of states is other than 2.

m Valid point types for DCMLC are:
0 = BI point
1 = Multi BI (two binary inputs connected to an IBN FM module in the FM slot)

n Ignored if the point type is BI.

o Ignored if the point type is Multi BI.

p Ignored if local contact wired 0 is Yes.

Continued on next page . . .

Multistate Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
TIMER	O						
		O	heavy equipment delay	integer	0-255	[5]	
		O	minimum On time	integer	0-255	[1]	q
		O	minimum Off time	integer	0-255	[0]	q
		O	max. starts per hour	integer	0-255	[255]	q r
FEEDBACK	O						
		R	feedback system name	name		[null]	
		R	feedback object name	name		[null]	
		CO	feedback closed for On	Boolean	Y/N	[Y]	j
REPORT	O						
		O	auto dial-up flag	Boolean	Y/N	[N]	
		O	point history flag	Boolean	Y/N	[Y]	
		CO	save pt. history flag	Boolean	Y/N	[N]	s
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	0-255	[0]	t
		O	normal report type	integer	0-6	[0]	u
		O	alarm report type	integer	0-6	[0]	u
		O	override report type	integer	0-6	[0]	u

Notes:
- j Ignored if number of states is other than 2.
- q Ignored if DCDRHW and local control = Y.
- r 0 means unlimited starts per hour.
- s Ignored if the point history flag is No.
- t Ignored if the corresponding report type is 0.
- u Valid report types (0-6) are:
  0 = no report
  1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
  5 = followup
  6 = status

Continued on next page . . .

Multistate Output Software Object (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
LOAD	O						
		R	load group system name	name		[null]	
		R	load group object name	name		[null]	
		O	priority	integer	1-4	[3]	
		O	load locked	Boolean	Y/N	[N]	
		R	load rating state 1	long	1-999999		
		O	load rating state 2	long	1-999999	[0]	
		O	load rating state 3	long	1-999999	[0]	
		O	shed state	integer	0-2	[0]	
		R	min. release time (min)	integer	1-255		
		R	max. shed time (min)	integer	1-255		
COMFORT	O						
		R	comfort system name	name		[null]	
		R	comfort object name	name		[null]	
		R	min. shed time (min)	integer	1-255	[0]	

Example

```
MSO “LOBBY”,“BOLT”,“Expanded_ID”,3,Y
HARDWARE “NC1HW”,“XRL4”,“HWLcS”,“HWLcO”
XMHW 4,1,252
XMLC 1,32,24,Y,N
UNITS “OFF”,“LOW”,“HIGH”
REPORT N,Y,N,N,2,1,1,1
```

Semantic Rules

MSO — The system name must already exist and be on the NCM for which the file is being compiled. The object name must be unique.

HARDWARE

The system\object must already exist, match the selected hardware type (for example, DCMHW, XMHW), and be on the same NCM.

If the hardware type is DCMHW or XMHW with XM Type 2 (XRM) or 3 (XRL/XRE), then the number of states must be 2.

If local contact is Yes, the local contact system\object must be specified, and they must exist, match the hardware type selected for the local contact, and be on the same NCM.

DCDRHW Only the XTM device type supports more than two states.

The hardware reference must be a valid Item in the hardware model of the specified device type and must be valid for an MSO with the specified number of states. All hardware references used by the MSO must be free.

DCM140HW

The slot number must be free. The pulse duration must be divisible by 20.

DCMHW The slot number must be free. The pulse duration must be divisible by 20.

LONHW The hardware reference must be valid for the specified hardware device type.

The hardware reference must be free.

The hardware reference must be valid for an MSO with the specified number of states.

IMPORTANT:	This information applies to objects mapped to DCDRHW and LONHW. Though the following is not checked by the DDL compiler, it is important for you to consider. If you are mapping a CS object attribute and a standard object to the same hardware reference, make sure the Override and Adjust flags are set to NO (false) for the CS object attribute. This is to ensure that there is only one command path to the hardware reference.

XMHW The pulse duration must be divisible by 12.

See the table below to determine the valid number of states and slot numbers for the XMHW:

XM Type	Permissible Number of States	Slot Numbers:	
		2 State	3 or 4 State
XRL 2x2 **XRM 2x2**	2 or 3	3, 4, 7, and 8	1 and 5
XRL 2x3 **XRM 2x3**	2 or 4	4 and 8	1 and 5

For 2 states, the slot number must be free.

For 3 states, the slot number and the slot number + 1 must be free.

For 4 states, the slot number, the slot number + 1, and the slot number + 2 must be free.

DCDRLC The Hardware Reference must be a valid Item in the hardware model of the specified Device type and must be valid for a binary input. All Hardware References used by the local contact must be free. If local contact wired 0 is Yes (XTM only), then only odd Item numbers are permitted for the Hardware Reference, and the next higher Item number must be free for the wired 0.

DCM140LC The slot number must be free. For Multi BI, the slot and subslot pair must be free.

DCMLC The slot number must be free. For Multi BI, the slot and subslot pair must be free.

XMLC Debounce filter must be divisible by 12.

See the table below to determine valid slot numbers for the XMLC:

XM Type	Local Contact Wired 0	Slot Numbers
XBN	N	1-32
XBN	Y	1-31, odd numbered
Not XBN	N	1-8
Not XBN	Y	1-7, odd numbered

If local contact wired 0 is “Yes”, the slot number and the slot number + 1 must be free; otherwise, the slot number must be free.

UNIT The initial value must be less than the number of states.

FEEDBACK The system\object must already exist (WARNING).

The software object must be MSI or MSD.

LOAD The load group reference must already exist (WARNING).

The load group reference must be a DLLR load group object.

The minimum release time must be greater than or equal to the minimum on time. (The minimum on time is a parameter of the TIMER subkey.)

If there are multiple states, the load rating of the second state must be greater than the load rating of the first state, and the load rating of the third state must be greater than the load rating of the second state.

COMFORT The COMFORT subkey can be used only if the LOAD subkey is also used.

The referenced comfort object must already exist (WARNING).

The referenced comfort object must be a BI, BD, AI, or AD.

The minimum shed time must be less than the maximum shed time.

NC File
N2OPEN Keyword

N2Open Device (AHU, MIG, NDM, PHX, UNT, VAV, VMA, VND) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
N2OPEN							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	hardware address	integer	1-255		
		O	N2 trunk number	integer	1-2	[1]	
		O	poll priority	integer	0-3	[3]	a
		R	device type	char (8)			b, c
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	

Notes:
a For poll priority: 0 = highest (most often polled)
3 = lowest (least often polled)
b Device type must be AHU, MIG, NDM, PHX, UNT, VAV, VMA, or VND.
c Do not define an XT that exists on the XT Bus as an N2 open device.

Example

```
N2OPEN "NC1N2","AHU22","AHU address 22"
ADDRESS 22,1,1,"AHU"
GRAPHICS 0,0
REPORT N,N
```

Semantic Rules

N2OPEN The system name must already exist. The object name must be unique.

ADDRESS The N2 trunk number must be defined as an N2 in the NCM port definition.

The address must not be used by another N2 device on the same trunk.

A warning occurs if you use 0 as the N2 address. (This is because an address of 0 can cause a problem in ASCs with firmware prior to A03, B03, or C03.)

NC File
PIDL Keyword

PID Loop Software Object Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
PIDL							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
PIDLNUM	O						
		O	PID loop number	integer	1-20	[1]	
REPORT	O						
		O	auto dial-up	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		CO	alarm message #	integer	1-255	[0]	a
Note: a Ignored if the Alarm report type is 0.							
Continued on next page . . .							

PID Loop Software Object Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		O	normal report type	integer	0-6	[0]	
		O	alarm report type	integer	0-6	[0]	
		O	override report type	integer	0-6	[0]	
UNITS	O						
		O	engineering units	char (6)		[DEGF]	
		O	decimal display posit.	Integer	0-3	[1]	
		O	function	Integer	1-3	[1]	b
FILTER	O						
		O	filter weight	fp	>=1.0	[1.0]	
CALIB	O						
		O	loop period	integer	1-32767	[5]	
		O	proportional band	fp		[20.0]	
		O	integral time	fp	>=0.0	[60.0]	
		O	derivative weight	fp	>=0.0	[0.0]	
		O	deadband	fp	>=0.0	[1.0]	
		O	hysteresis	fp	0.0-100.0	[0.0]	
		O	tune noise band	fp	>=0.0	[4.0]	
		O	tune change factor	fp	0.0-1.0	[0.0]	
RELIAB	O						
		O	unrel. default respon.	fp		[0.0]	
		O	unrel. default selector	Boolean	Y/N	[N]	
INPUT	O		(from zero to six inputs can be defined)				
		O	reference flag	Boolean	Y/N	[N]	
Note: b Valid functions are: 1 = sum 2 = minimum 3 = maximum							
Continued on next page . . .							

PID Loop Software Object Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
		CO	value	fp		[0.0]	c
		CR	system name	name		[null]	d
		CR	object name	name		[null]	d
		O	scalar	fp		[0.0]	
SETPOINT	O						
		O	reference flag	Boolean	Y/N	[N]	
		CO	value	fp		[55.0]	c
		CR	system name	name		[null]	d
		CR	object name	name		[null]	d
OFFSET	O						
		O	reference flag	Boolean	Y/N	[N]	
		CO	value	fp		[0.0]	c
		CR	system name	name		[null]	d
		CR	object name	name		[null]	d
LOSAT	O						
		O	reference flag	Boolean	Y/N	[N]	
		CO	value	fp		[0.0]	c
		CR	system name	name		[null]	d
		CR	object name	name		[null]	d
HISAT	O						
		O	reference flag	Boolean	Y/N	[N]	
		CO	value	fp		[100.0]	c
		CR	system name	name		[null]	d
		CR	object name	name		[null]	d
AUXIN	O						
		O	reference flag	Boolean	Y/N	[N]	
		CO	value	fp		[0.0]	c
		O	aux. enable	Boolean	Y/N	[N]	e
		CR	system name	name		[null]	d
		CR	object name	name		[null]	d

Notes:
- c Ignored if the reference flag is Yes.
- d If the reference flag is Yes, this parameter is required. If the reference flag is No, this parameter is ignored.
- e If Y, then auxiliary input is passed out of auxiliary switch.
  If N, the PID algorithm output is passed out of auxiliary switch.

Continued on next page . . .

PID Loop Software Object Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
HLIN	O						
		O	reference flag	Boolean	Y/N	[N]	
		CO	value	fp		[0.0]	c
		O	selector type high	Boolean	Y/N	[Y]	
		CR	system name	name		[null]	d
		CR	object name	name		[null]	d
OUTPUT	R		(from 1 to 8 outputs can be defined)				
		R	object type	char (8)	PIDL or AOD	[null]	
		R	system name	name		[null]	
		R	object name	name		[null]	
		R	attribute name	char (8)		[null]	f

Notes:
- c Ignored if the reference flag is Yes.
- d If the reference flag is Yes, this parameter is required. If the reference flag is No, this parameter is ignored.
- f The attribute name can be only one of 13 values:

VALUE	INP1VAL to INP6VAL
SETPOINT	OFFSET
HI_SAT_V	LO_SAT_V
AUX_IN	SEL_INP

Example

```
PIDL “AC3”,“T1”,,
HARDWARE “NC1-HW”, “DCM1”
PIDLNUM 1
INPUT Y,,“NC1”, “ROOMTEMP”
OUTPUT “AOD”,“NC1”,“ROOMVALVE”,“VALUE”
```

Semantic Rules

PIDL — The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist and must be a DCM or DCM140 on the same NCM.

PIDLNUM — The loop number must be free. For DCM101 (defined with DCM keyword), loop numbers 1 through 16 are available. For DCM140 (defined with DCM140 keyword), loop numbers 1 through 20 are available.

There will be a compile error if a PID loop number 17 through 20 is entered for a DCM101.

CALIB — The tune noise band > deadband.

INPUT — You can have from 0 to 6 of these.

INPUT, SETPOINT, OFFSET, HISAT, AUXIN, HLIN

If reference flag is Y, then the system\object must already exist (WARNING).

The system\object must be an AI on the same DCM as the PIDL.

OUTPUT — You can have from 1 to 8 of these.

The system\object must already exist (WARNING).

The system\object must be an AOD or PIDL on the same DCM as the PIDL.

For an AOD, the attribute must be VALUE.

For a PIDL, the attribute cannot be VALUE.

NC File
READER Keyword

Card Reader (READER) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
READER							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	
		O	alarm message number	integer	0-255	[0]	
		O	normal report type	integer	0-6	[0]	
		O	alarm report type	integer	0-6	[0]	
READERID	R						
		R	reader number	integer	1-16		
		R	reader type	integer	0-4		a
		R	card type	integer	0-5		b
		O	local reader ID	char (15)		[null]	

Notes:
a Reader types:
0 = DISABLED
1 = ACCESS
2 = IN READER
3 = OUT READER
4 = ALM READER

b Card types:
0 = NO CARD
1 = WIEGAND NON-NCRYPT
2 = WIEGAND NCRYPT
3 = BA FE NO PARITY
4 = MAG STRIPE
5 = BA FE WITH PARITY

Continued on next page . . .

Card Reader (READER) Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
DOORCHEK	R						
		O	no alarm on exit	Boolean	Y/N	[N]	
		O	anti-tailgate check	Boolean	Y/N	[Y]	
		O	anti-passback check	Boolean	Y/N	[Y]	
		O	facility code on backup	Boolean	Y/N	[Y]	
		O	PIN code on backup	Boolean	Y/N	[N]	
TIMECHEK	R						
		R	door access time	integer	0-25		
		R	door shunt delay	integer	1-255		
		R	reader active time zone #	integer	1-8		
		O	override time zone #	integer	0-8	[0]	c
		O	PIN suppress time zone #	integer	0-8	[0]	c
		O	anti-passback timer	integer	0-60	[0]	
Note: c The default 0 value means no time zone applies.							

Example

```
READER “AC-SYS”,“READER3”,“LOBBY READER”
HARDWARE “AC-SYS”, “D600-11”
GRAPHICS 3,7
READERID 3,2,4, “LOBBY”
DOORCHEK Y,Y,Y,Y,N
TIMECHEK 8,12,1,3,0,30
REPORT N,N,7,3,6
```

Semantic Rules

READER The system name must already exist. The object name must be unique.

HARDWARE

The hardware system/object must be an Access controller (D600), and must be on the same NCM.

READERID The card reader number must be unique for this Access controller.

TIMECHEK Door access time must be smaller than the door shunt delay.

The reader active time zone # must reference an existing time zone.

Override time zone # and PIN suppress time zone # must reference an existing time zone if they are defined as non-zero.

NC File SUBORDINATE
Keyword

Subordinate Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
SUBORDINATE							
		R	subordinate object type	char (8)		[null]	a
		R	subordinate input system	name		[null]	
		R	subordinate input object	name		[null]	
		R	MC system name	name		[null]	
		R	MC object name	name		[null]	
STATE0	O						
		O	delay time in seconds	long	0-86399	[0]	
		CR	command name	char (6)		[null]	b
		CR	command parameters string	char (50)		[null]	c
		CR	commanded attribute	char (8)		[null]	d
STATE1	O						
		O	delay time in seconds	long	0-86399	[0]	
		CR	command name	char (6)		[null]	b
		CR	command parameters string	char (50)		[null]	c
		CR	commanded attribute	char (8)		[null]	d

Notes:

a Valid object types are:

ACM	BO	DCM	FPU	MSI
AD	C210A	DCM140	JCB	MSO
AI	C260A	DLLR	LCD	N2OPEN
AOD	C260X	DSC-1000	LCG	PIDL
AOS	C500X	DSC8500	LON	READER
BD	CS	D600	MC	XM
BI	DCDR	FIRE	MSD	ZONE

b Required if delay is greater than 0.

c Required if command exists and has parameters.

d Required if command exists and command is an attribute command.

Continued on next page . . .

Subordinate Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
STATE2	O						
		O	delay time in seconds	long	0-86399	[0]	
		CR	command name	char (6)		[null]	b
		CR	command parameters string	char (50)		[null]	c
		CR	commanded attribute	char (8)		[null]	d
STATE3	O						
		O	delay time in seconds	long	0-86399	[0]	
		CR	command name	char (6)		[null]	b
		CR	command parameters string	char (50)		[null]	c
		CR	commanded attribute	char (8)		[null]	d

Notes:
b Required if delay is greater than 0.
c Required if command exists and has parameters.
d Required if command exists and command is an attribute command.

Example

```
SUBORDINATE "AD","WEST","HTGSPT","PNL-4","MC-4"
   STATE0 10,"ALARMS","2.2,5.5,7.7"
   STATE1 20,"WARNINGS","3.34,7,19.2,91.2"
   STATE2 0,"SET_AD","44.3,2"
   STATE3 120,"RELEASE"
SUBORDINATE "CS","AHUC","AHU-2","PNL-4","MC-4"
   STATE0 10,"STCSAN","31.2,2","AD_1"
   STATE1 60,"STCSBN","0,2","BI_1"
   STATE2 120,"STCSMS","2,5","MS_1"
```

Semantic Rules

SUBORDINATE — The system\object name must exist (WARNING) and be of the same type as specified by the subordinate object type.

The MC system\object must exist and must be of the type MC on the NCM being compiled.

A maximum of 50 subordinates can be defined per MC.

STATE0-3 — The command must be valid for the object type. (See the Command Table on the next page for a list of commands by object type.)

All required parameters for the command must be specified. All the command parameters are a single parameter of the DDL subkey STATE0, STATE1, STATE2, STATE3. All the command parameters are enclosed within a single set of double quotes. For example, in the first defined subordinate above, the following command and command parameters are specified for STATE2:

```
"SET_AD","44.3,2"
```

Following SET_AD are the two command parameters required by the command, value and priority. In this case, the SET_AD command adjusts the AD object to 44.3 at Priority 2.

All commands and their required parameters are listed, by object type, in the Command Table on the following pages. (Commands are also listed in the command tables at the end of object technical bulletins).

For commands to individual fire alarm points or zones, the attribute specifies the fire alarm point or zone to be commanded using the syntax #xxxx where xxxx is a one to four character Device ID.

Command Table			
Object Type	**Commands**	**Parameters**	**Comments**
Access Controller (D600)	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	AcresAlr	None	Access Control reset alarm.
	AcgloNor	None	Access Control global normal operation (release global access).
	AcgloAcc	None	Access Control global access override command.
	NoOKcard	None	Stop reporting of access granted operations.
	YsOKcard	None	Start reporting access granted operations.
Accumulator	LOC REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	ALARMS	Low limit	Set/change alarm parameters.
		High limit	Use DEFAULT to specify default value; use DELETE to delete defined value.
		Differential	
	WARNINGS	Setpoint	Set/change warning parameters.
		Normalband	Either **both** setpoint and normalband must be in the command with specified values or both must be absent.
		Feedback time	Use DEFAULT to specify default value; use DELETE to delete defined value.
		Differential	
Analog Data	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	ALARMS	Low limit	Set/change alarm parameters.
		High limit	Use DEFAULT to specify default value; use DELETE to delete defined value.
		Differential	
	WARNINGS	Setpoint	Set/change warning parameters.
		Normalband	Either **both** setpoint and normalband must be in the command with specified values, or both.

Continued on next page . . .

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
		Feedback time	Must be absent. Use DEFAULT to specify.
		Differential	Default value; use DELETE to delete defined value.
	SET_AD	Value, Priority (2 or 3)	At Priority 2, override value. At Priority 3, replace value.
	RELEASE	None	Release Priority 2 value.
Analog Input	LOC REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
Analog Input	ALARMS	Low limit	Set/change alarm parameters.
		High limit	Use DEFAULT to specify default value; use DELETE to delete defined value.
		Differential	
	WARNINGS	Setpoint	Set/change warning parameters.
		Normalband	Either **both** setpoint and normalband must be in the command with specified values or both must be absent.
		Feedback time	
		Differential	Use DEFAULT to specify default value; use DELETE to delete defined value.
Analog Output Digital	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	SET_AOD	Value	Override output at Priority 2.
	RELEASE	None	Release Priority 2 value.
Analog Output Setpoint	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	SET_AOS	Value, Priority (2 or 3)	At Priority 2, override value. At Priority 3, replace value.
	RELEASE	None	Release Priority 2 value.
	RELEASE 3	None	Release Priority 3 value.
Continued on next page . . .			

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
Binary Data	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	SET_BD	Value, Priority (2 or 3)	At Priority 2, override value. At Priority 3, replace value.
	RELEASE	None	Release Priority 2 value.
	UNLATCH	None	Reset latched condition.
Binary Input	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	UNLATCH	None	Reset latched condition.
Binary Output	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	REL_PRI	Priority	Release at specified priority.
	START	Priority	Output State 1 at priority.
	STOP	Priority	Output State 0 at priority.
C210A	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	ST210AAN*	Value	Override value of attribute at Priority 2.
	ST210ABN*	Value	Override value of attribute at Priority 2.
	ST210ASP*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, replace value of attribute.
	REL_210A*	Priority (2 or 3)	Release at specified priority.
	* Must specify attribute.		
Continued on next page . . .			

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
C260A	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	ST260AAN*	Value	Override value of attribute at Priority 2.
	ST260ABN*	Value	Override value of attribute at Priority 2.
	ST260ASP*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, replace value of attribute.
	REL_260A*	Priority (2 or 3)	Release at specified priority.
	* Must specify attribute.		
C260X	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	ST260XAN*	Value	Override value of attribute at Priority 2.
	ST260XBN*	Value	Override value of attribute at Priority 2.
	ST260XSP*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, replace value of attribute.
	REL_260X*	Priority (2 or 3)	Release at specified priority.
	* Must specify attribute.		
C500X	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	ST500XAN*	Value	Override value of attribute at Priority 2.
	ST500XBN*	Value	Override value of attribute at Priority 2.
	ST500XSP*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, replace value of attribute.
	REL_500X*	Priority (2 or 3)	Release at specified priority.
	* Must specify attribute.		
Continued on next page . . .			

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
Control System	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	STCSAN*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, adjust value of attribute.
	STCSBN*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, adjust value of attribute.
	STCSMS*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, adjust value of attribute.
	REL_CS*	Priority (2 or 3)	At Priority 2, release override of attribute. At Priority 3, release adjust of attribute.
	* Must specify attribute. Must know (from the software model on which the CS object is based) whether the attribute is defined to be adjustable. For STCSAN, STCSBN, and STCSMS commands, must know which attributes are valid for each command.		
DCDR	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
DCM	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
DCM140	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
Demand Limiting and Load Rolling	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	LOC_LOAD	System\object	Don't allow load to be shed.
	UNL_LOAD	System\object	Allow load to be shed.
	DL_MON	None	Monitor only - inhibits further load shedding by DL.
	DL_SHED	None	Enable load shedding by DL.
	DL_TARGT	Value, level (0-3)	Change demand limit target.
	LR_MON	None	Monitor only - inhibits further load shedding by LR.
Continued on next page . . .			

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
	LR_SHED	None	Enable load shedding by LR.
	LR_TARGT	Value, level (0-3)	Change load rolling target.
	RESETPER	None	Set current demand and consumption to 0.
DSC8500	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
DSC-1000	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
Expansion Module (XM)	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
FIRE	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	F_RESET[2]	None	Reset a fire panel.
	F_DISABL[1]	None	Fire point or zone disable.
	F_ENABLE[1]	None	Fire point or zone enable.
	F_PT_ON[2]	None	Command fire control point on.
	F_PT_OFF[2]	None	Command fire control point off.
FPU	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
JC-BASIC	PRC_ENA	None	Enable process operation.
	PRC_DIS	None	Disable process operation.
	TRIGGER	None	Force a JC-BASIC process to run.
Lighting Controller	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
Lighting Control Group	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	TIMED_ON	Hours (optional)	Timed override for hours. Use DEFAULT to specify default value.

1 Commands to a fire zone within a fire panel are limited to F_DISABL and F_ENABLE.

2 Supported only on UL864 Fire NCM type when defined on the same NCM as the Fire panel.

Continued on next page . . .

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
	ON	None	Turn group on, overrides LC schedule.
	OFF	None	Turn group off, overrides LC schedule.
	RELEASE	None	Release JC-BASIC control.
LON	LOC TRIG	None	Disable Process Triggering.
	UNL TRIG	None	Enable Process Triggering.
MC	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	RELEASE	None	Release Priority 2 value.
	SET_MC	Value, Priority (2 or 3)	At Priority 2 override value.
			At Priority 3 replace value.
	UNLATCH	None	Reset latched condition.
MSD	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	RELEASE	None	Release Priority 2 value.
	SET_MSD	Value, Priority (2 or 3)	At Priority 2 override value At Priority 3 replace value
	UNLATCH	None	Reset latched condition.
MSI	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	UNLATCH	None	Reset latched condition.
MSO	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	REL_PRI	Priority	Release at specified priority.
	SET_MSO	Value, Priority (2 or 3)	At Priority 2, override value.At Priority 3, replace value.
N2OPEN	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
PID Controller	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
Continued on next page . . .			

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
PID Loop	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	AUX_ENA	None	Allow Aux input to pass through PID loop.
	AUX_DIS	None	Do not allow Aux input to pass through PID loop.
	SET_PIDL*	Value, Priority (2 or 3)	At Priority 2, override value of attribute. At Priority 3, replace value of attribute.
	REL_PIDL*	None	Release Priority 2 value of attribute.
	* Must specify attribute. Valid attributes are: INP1VAL, INP2VAL, INP3VAL, INP4VAL, INP5VAL, INP6VAL, SETPOINT, OFFSET, HI_SAT_V, LO_SAT_V, AUX_IN, SEL_INP		
	STARTUP	None	Restart PID algorithm.
READER[1]	LOC_REP	None	Disable COS annunciation.
	UNL_REP	None	Enable COS annunciation.
	LOC_TRIG	None	Disable process triggering.
	UNL_TRIG	None	Enable process triggering.
	secure	None	Lock (secure) door at time zone priority.
	access	None	Unlock (access) door at time zone priority.
	openDr	None	Unlock door temporarily to admit one person.
	secDrOvr	None	Lock (secure) door at override priority (may not be supported by all access controller versions).
	accDrOvr	None	Unlock (access) door at override priority (may not be supported by all access controller versions).
	relDrOvr	None	Release override priority command (may not be supported by all access controller versions).
Trend	BEG_TRND	None	Start trend sampling for attribute. Must specify attribute.
	END_TRND	None	Stop trend sampling for attribute. Must specify attribute.
Totalization	BEG_TOT	None	Start tot sampling for attribute. Must specify attribute.
Continued on next page . . .			

Command Table (Cont.)			
Object Type	**Commands**	**Parameters**	**Comments**
	END_TOT	None	Stop tot sampling for attribute. Must specify attribute.
	RES_TOT	Value	Reset totalized value of specified attribute. Must specify attribute. Use DEFAULT to specify default value.
ZONE [1,2]	LOC_REP	None	Disable COS annunciation
	UNL_REP	None	Enable COS annunciation
	LOC_TRIG	None	Disable process triggering
	UNL_TRIG	None	Enable process triggering
	F_RESET[3]	None	Reset a fire panel (to IFC fire panel only)
	F_DISABL[3]	None	Fire point or zone disable (to IFC fire panel object or fire zone object)
	F_ENABLE[3]	None	Fire point or zone enable (to IFC fire panel object or fire zone object)
	F_PT_ON[3]	None	Command fire control point on (to IFC fire panel object or fire zone object)
	F_PT_OFF[3]	None	Command fire control point off (to IFC fire panel object or fire zone object)

1 Fire panels must be defined as follows for correct operation:
For Fire NCM type: Trunk 1 = N2 connection for fire panels
Trunk 2 = Fire Net connection for fire panels
Other N2 devices on Trunk 2 work normally.

2 Commands to a fire zone within a fire panel are limited to F-DISABL and F_ENABLE. Refer to *Metasys Intelligent Fire Network Technical Bulletin (LIT-448196)* for details.

3 Supported only on UL864 Fire NCM type when defined on the same NCM as the fire panel.

Timezone Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
TIMEZONE							
		R	D600 system name	name			
		R	D600 object name	name			
		R	time zone record number	integer	1-8		
MON	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b
TUE	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b
WED	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b
THU	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b

Notes:
a 0-8 of these subkeywords are allowed.
b Time zone is disabled starting at specified time if this parameter is [Y].
c Time zone time2 defaults are undefined, which equates to an FFFF in the database.

Continued on next page . . .

Timezone Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
FRI	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b
SAT	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b
SUN	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b
HOL	O						a
		R	time1	integer	0-2359		
		R	disable1	Boolean	Y/N		b
		O	time2	integer	0-2359		c
		O	disable2	Boolean	Y/N	[N]	b

Notes:
a 0-8 of these subkeywords are allowed.
b Time zone is disabled starting at specified time if this parameter is [Y].
c Time zone time2 defaults are undefined, which equates to an FFFF in the database.

Example

```
TIMEZONE “AC-SYS”,“D600-12”,3
MON 0000,Y,0800,N
MON 1805,Y
TUE 0000,N,1805,Y
WED 0000,N,1805,Y
THU 0000,N,1805,Y
FRI 0000,N,1805,Y
SAT 0000,N,1230,Y
SUN 0000,N
HOL 0000,N,2300,Y
```

Semantic Rules

TIMEZONE The Access controller (D600 system/object) must exist and be on the same NCM as the TIMEZONE database.

MON...HOL The time zone time must be entered as HHMM, where HH is hours (0-23) and MM is minutes (0-9).

Time zone times must alternate enable and disable functions in a day, which means an enable cannot be followed by another enable in the same day. Likewise, a disable cannot follow a disable.

Time zone times are entered as pairs with one exception. Only the last time1 entered on a given day may stand alone with no paired time. The enable or disable state will carry over through midnight to become the initial state of the next day. A new day can start with an enable or disable state regardless of the initial state carried over from the previous day.

Time zone pairs must be entered in time order within a given day.

For each MON, TUE,...HOL subkey, if it is used, the first enable/disable time must be 0000.

NC File XM
Keyword

Expansion Module (XM) Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
XM							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (24)		[null]	
ADDRESS	R						
		R	device address	integer	0-255		
		O	N2 trunk number	integer	1-2	[1]	
		R	XM type	integer	1-3		a
		O	poll priority	integer	0-3	[1]	b
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[N]	
		O	comm. disable	Boolean	Y/N	[N]	

Notes: a Valid XM types are:

1 = XBN

2 = XRM

3 = XRL or XRE

4 = XRL 2x2

5 = XRL 2x3

6 = XRM 2x2

7 = XRM 2x3

Device types 4, 5, 6, and 7 refer to international models. Because DDL will not error out a 4, 5, 6, or 7, a mistake is not indicated if any of these values is entered.

b For poll priority: 0 = highest (most often polled)

3 = lowest (least often polled)

Example

```
XM “NC1-HW”,“XBN1”,“XBN-in-EN-1”
   ADDRESS 2,1,1,1
   GRAPHICS 0,0
   REPORT N,N
XM “NC1-HW”,“XRL4”,“Left-XRL-in-EN-2”
   ADDRESS 4,1,3,1
```

Semantic Rules

XM	The system name must already exist. The object name must be unique.
ADDRESS	The N2 trunk number must be defined as an N2 port on the NCM. The address must not be used by another N2 device on the same trunk. A warning occurs if you use 0 as the N2 address, because an address of 0 can cause a problem in ASCs with firmware prior to A03, B03, or C03.)

NC File ZONE
Keyword

Fire Zone Software Object Definition							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
ZONE							
		R	system name	name			
		R	object name	name			
		O	expanded ID	char (20)		[null]	a
HARDWARE	R						
		R	hardware device system	name			
		R	hardware device object	name			
GRAPHICS	O						
		O	symbol number	integer	0-32767	[0]	
		O	operator instruction	integer	0-32767	[0]	
IFC2020	R						
		R	zone number	integer	1-240		
		O	interlock statement	char (70)		[( )]	b
		O	annunciator panel	integer	0-32	[0]	
		O	annunciator point	integer	0-64	[0]	

Notes:
- a Zone expanded ID is limited to the 20 characters that fit in the fire panel hardware. When downloaded, all characters are converted to uppercase.
- b For information on the interlock statement, see the *IFC-1010/2020 Programming Technical Bulletin (LIT-448060)*. If this is a reverse activated zone, an equation must be entered.

Continued on next page . . .

Fire Zone Software Object Definition (Cont.)							
Keyword	**KWOPT**	**R/O/C**	**Parameter**	**Type**	**Restrict**	**Def**	**Note**
REPORT	O						
		O	auto dial-out	Boolean	Y/N	[Y]	
		O	comm. disable	Boolean	Y/N	[N]	
		CO	normal message #	Integer	0-255	[0]	c
		CO	alarm message #	Integer	0-255	[0]	c
		CO	disable local message #	Integer	0-255	[0]	c
		O	normal report type	Integer	0-6	[6]	d
		O	alarm report type	Integer	0-6	[1]	d
		O	disable local report type	Integer	0-6	[2]	d
		O	trouble report type	integer	0-6	[3]	d

Notes:
c Ignored if corresponding report type is 0.
d Valid report types (0-6) are:
0 = no report
1, 2, 3, or 4 = Critical1, Critical2, Critical3, or Critical4
5 = followup
6 = status

Note: If you create a Fire ZONE using DDL and download the NCM, none of the ZONE data are downloaded to the IFC-2020. When you use DDL to define ZONE objects, you also must configure the IFC-2020 to reflect the same zone definitions and same last forward zone. You may need to add or change interlock statements.

Example

```
ZONE “IFC2ZONE”, “ZONE10”
HARDWARE “FIRE”,“IFC2”
GRAPHICS 90,105
IFC2020 10,,12,44
REPORT Y,N,56,55,57,6,1,2
```

Semantic Rules

ZONE — The system name must already exist. The object name must be unique.

HARDWARE

The system\object must already exist and must be a FIRE object on the same NCM.

IFC2020 — The number must be free.

If the zone is forward activated, the zone number must be less than or equal to the Last Forward Zone Number defined for the FIRE object.

Reverse activated zones must be numbered above the Last Forward Zone number. The Interlock statement cannot be defaulted in this case.

Decompiler

Overview

The DDL decompiler acts as a reverse compiler: it reads the archive database on the Operator Workstation and creates DDL source code files from this information. It places the source code files it generates in the appropriate DDL subdirectories and gives them a .UND extension to differentiate them from .DDL source code files.

The DDL decompiler runs as a program under DOS. The decompiler is installed along with the DDL compiler.

Purpose

The decompiler's main purpose is to produce DDL source code files from the archive database.

Once a database has been compiled and downloaded, it can be modified with several tools; DDL and GPL modify the archive database directly, and online generation modifies the online database, which can then be uploaded to update the archive database. The decompiler allows you to inspect this modified archive database in DDL source code format, giving you a consistent and comprehensive view.

The decompiled database can be recompiled using the DDL compiler.

Input to the Decompiler

As input, the DDL decompiler uses information from an existing archive database on an Operator Workstation. The archive database can be created and modified by any of the Metasys database generation tools, including DDL, GPL, and online generation (in conjunction with an upload).

The decompiler decompiles *only* those portions of the archive database that can be recompiled with the DDL compiler.

IMPORTANT:	A JCB keyword decompiles for each process in the archive database. However, when recompiled with the DDL compiler, process objects do not remain intact and therefore need to be recompiled at the workstation (or retranslated with GPL). For more information, see the *Introduction* to this manual.

The Release 8.0 decompiler can only decompile a database if it is at Release 6.0, 7.0, or 8.0 (created at Release 6.0, 7.0, or 8.0 or converted to Release 6.0, 7.0, or 8.0). If the database is not at Release 6.0, 7.0, or 8.0, the decompiler will notify you that a conversion is necessary before a decompile can take place. The Release 8.0 decompiler can decompile databases created at any release, as long as they have been converted to Release 6.0, 7.0, or 8.0.

When executing the decompiler, you have a number of options for choosing which portions of the archive database will be decompiled. For example, you can select the entire FMS, an entire network, or only one NC database. The section *Executing the Decompiler* discusses these options.

Output From the Decompiler

As output, the decompiler creates source code files in standard DDL format. These are ASCII files that can be edited with any text editor. The decompiler gives these files a .UND extension to differentiate them from .DDL source code files.

For more information on the appearance of .UND files, see *What Decompiled Files Look Like*, later in this section.

How the Decompiler Works

The DDL decompiler decompiles one FMS at a time. It decompiles the portion of the FMS you specify when you start the decompile. For example, it decompiles the archive database for an entire FMS, or for only one NC.

The decompiler creates only one source code file at a time. If a decompile will result in more than one source code file, the decompiler completes one entire file before starting another.

Guidelines For Effective Operation

Before you use the decompiler, consider the following:

- The Release 8.0 compiler and decompiler will only work on Release 6.0, 7.0, or 8.0 databases. If the compiler or decompiler is invoked and the database is at an earlier release, the compile or decompile will not occur and you will be instructed to convert the database. See the *Operator Workstation (OWS) Technical Bulletin* for instructions. After the conversion is complete, you are free to compile or decompile the converted database.
- The decompiler decompiles only what the current version of DDL can recompile.
- Make sure you perform an upload (NC and Global) before decompiling. The decompiler reads the archive database for NCs and Globals; however, all online changes must be uploaded for the decompiler to have access to them.
- A JCB keyword decompiles for every process in an NC. However, if you recompile the DDL source code, you must also recompile processes at the workstation (or retranslate with GPL), since process objects do not remain intact after a compile.

Executing the Decompiler

This section tells you how to execute the DDL decompiler.

IMPORTANT:	Before you start the decompiler, make sure you do an Upload of *all* databases you intend to decompile. This will ensure the archive database on the Operator Workstation is the most up-to-date version of the database, and includes all online changes.

Command Line

To execute the decompiler, you use one of several UNDDL command options. These options allow you to specify the portion of the archive database you want to decompile. You also specify whether you want the decompiler to overwrite existing .UND files.

All commands are case-insensitive.

After typing the command, press Enter.

Table 1 explains the command line options.

Command	Function	Required Parameters
UNDDL ?	Displays decompiler help screen	None
UNDDL NET	Decompiles Workstation Network/Port file	None
UNDDL MODEL	Decompiles Model file	None
UNDDL GLOBAL	Decompiles Global file for specified network	Network Name
UNDDL NC	Decompiles one NC file for specified network	Network Name and NC Name
UNDDL FMS	Decompiles entire archive database: Workstation Network/Port, Model, and all Global and NC files	None
UNDDL ONENET	Decompiles all files for specified network: Global and all NC files	Network Name

Table 1: Decompiler Command Line Options

For example, to decompile the archive database for a network called Central, type the following at the DOS prompt:

```
UNDDL ONENET CENTRAL
```

This decompile would produce one Global file and one NC file for each NC defined in the Global file.

To decompile the entire archive database, type the following at the DOS prompt:

```
UNDDL FMS
```

This decompile would produce one Workstation Network/Port file, one Model file, one Global file for each network defined in the Workstation Network/Port file, and one NC file for each NC defined in each Global file.

To decompile a single NCM's archive database, type the following at the DOS prompt:

```
UNDDL NC EASTNET NC2
```

This decompile would produce one NC file named NC2.UND.

Overwriting Files

To automatically overwrite files with the same name and same .UND extension, type `/Y` at the end of the command line. Precede the `/Y` with a space.

This does not overwrite .DDL files. It overwrites only those .UND files with the same name as the newly created .UND files.

For example, to automatically overwrite existing .UND files during a decompile of a network called Central, type the following at the DOS prompt:

```
UNDDL ONENET CENTRAL /Y
```

If you do not type `/Y`, and the decompiler finds a .UND file with the same name as the one it is about to create, it asks whether you want to overwrite the file. If you type Y (Yes), the existing file is overwritten. If you type N (No), the decompile of that file cancels, and the decompiler goes on to the next file.

Note: If you do not want to overwrite existing .UND files because you want to save an old version of the decompile, use DOS to rename the old files before you execute the decompiler.

Procedure

To execute the decompiler:

1. Do an Upload of *all* databases (global and NC) you intend to decompile.
2. Make sure the FMSDOS environment variable points to the WIN.INI file for the database you are decompiling.

   For example, if the WIN.INI file for the FMS database you are decompiling resides in the C:\FMS\DATA directory, set the FMSDOS environment variable by typing the following at the DOS prompt:

   ```
   SET FMSDOS = C:\FMS\DATA
   ```

   If the WIN.INI file for the archive database you are decompiling resides in the C:\PROJECT\90120001 directory, set the FMSDOS environment variable by typing the following at the DOS prompt:

   ```
   SET FMSDOS = C:\PROJECT\90120001
   ```

   For more information on the FMSDOS environment variable and WIN.INI file, see the *Compiler* section, under *Operator Workstation Disk Layout for DDL*.
3. Type the UNDDL command that specifies the type of decompile you want to perform, and press Enter.

   To see the UNDDL help screen, which lists the command line options, type `UNDDL ?`.

   If you type an incomplete command line, for example, `UNDDL NC CENTRAL` without the NC name, the decompile does not execute, and a list of valid NC names appears. You can then note the NC name and type the command line again.

   Similarly, if you type the name of a network or NC that does not exist, the decompile does not execute, and a list of valid network or NC names appears on the screen.

   If you type `/Y` after the command, any .UND files with the same name are automatically overwritten, and the screen displays the name of these files as they are being decompiled and overwritten. If there are no duplicate files, the decompiler displays the names of the new files as they are being created.

If you do not type `/Y` after the command, and a .UND file with the same name already exists, the screen displays the name of the file, and asks whether you want to overwrite it. For example, the following would appear if a file called NET.UND exists:

```
decompiling archive database to "FMSdata"\DDL\NET.UND
Warning, file "FMSdata"\DDL\NET.UND exists
Overwrite file? (Y/N)
```

If you type N (No), the decompile for the NET.UND file cancels, and the decompiler goes on to the next file. If you type Y, the decompile continues, and the screen displays the name and location of the file being overwritten.

4. Wait for the decompile to finish. The decompile takes slightly less time than it would take to compile a comparable database.

Screen Output

Upon execution, the decompiler displays a copyright message, followed by the name of the .UND file that is being created as a result of the decompile, and this file's location. For example, during decompile of a NET file, the following line appears on the screen:

```
decompiling archive database to "FMSdata"\ddl\net.und
......
```

The screen displays information about the status and result of the decompile. During the decompile, it displays `.` (dot) as each portion of the database is successfully decompiled. It displays `*` (asterisk) for every error. The `.` and `*` display at different speeds, depending on which keyword is being decompiled.

When the decompile is complete without errors, the following message appears:

```
UNDDL Decompile complete with no errors.
```

Error Notification

If errors are found during the decompile, the screen tells you how many. For example, the following message is displayed after the decompile if eight total errors, in all decompiled files, are found:

```
UNDDL Decompile complete with 8 errors.
```

The decompiler adds specific, explanatory error messages to the decompiled .UND source code files. Errors are explained later in this section, under *Error Handling*.

Exit Code

The decompiler returns a DOS exit code at the end of the decompile. The exit code is 0 if no errors are found during the decompile, and 1 if errors are found. This exit code is useful if you are using DOS batch streams to run the decompiler.

Where Decompiled Files Go

The decompiled files are given a .UND extension, and placed in the following file structure. This is the suggested file structure as described in the *Compiler* section, under *Operator Workstation Disk Layout for DDL*.

```
"FMSdata"\DDL
       \net.UND
       \model.UND
       \[NETWORK1]
              \global.UND
              \[NC1].UND
              \[NC2].UND
              \[basement].UND
       \[NETWORK2]
              \global.UND
              \[NC1].UND
              \[NC2].UND
              \[basement].UND
```

"FMSdata" refers to the FMSdata parameter specified in the WIN.INI file.

The names of Net, Model, and Global files are fixed. The bracketed names of NC files (e.g., [NC1], [basement]) represent user-defined NCM names. The bracketed names of network directories (e.g., [NETWORK1]) represent the names of the networks as defined in the network database.

If the directories are not present on the workstation performing the decompile, the decompiler creates them.

What Decompiled Files Look Like

The header for a .UND file is commented out with preceding asterisks, and includes:

- copyright information
- the date and time of decompile
- the path set by the FMSdata parameter in the WIN.INI file, for example:

```
archive decompiled from:
c:\fms\data\win.ini FMSDATA = c:\fms\data
```

Following the header is the DDL source code in standard DDL format. The @ file keyword (@NET, @MODEL, @GLOBAL, @NC) is followed by the main keywords.

For the Workstation Network/Port Configuration, Model, and Global files, the keywords are in an order that will allow a recompile of the file. That is, keywords that may be referenced by other keywords come first.

For the NC file, the keywords are in the order determined by the object.DOB file. The object.DOB file specifies either the order in which the objects were defined, or the order set by the Change Object Order function on the Operator Workstation.

To make decompiled files easy to read:

- Two blank lines are inserted between the file keyword and the first main keyword.
- One blank line is inserted between main keywords. (Subkeywords are not separated by blank lines.)
- Each subkeyword is indented four spaces.
- Each comma is followed by a space.
- All lines that could exceed 80 characters (if parameters are at their maximum lengths) are split with a \ (continuation character). For each keyword, the \ splits the line in the same place, regardless of whether the parameters are at their maximum lengths.

Example Workstation Network/Port File

Here is an example of a decompiled Workstation Network/Port file (without the copyright header).

```
@NET

PORT "LPT1", 3

NET "XYZ-BLDG", "XYZ-Building", "PC1"
    N1DIRECT 1, 100
*
****************************************
*          @NET Keyword Summary
*
*   PORTs 1       NETs  1
*
*   Total Keywords = 2
****************************************
*   no errors decompiling.
****************************************
```

The keyword summary lists the number of each type of keyword decompiled, and the total number of keywords. The error comment following the summary indicates the total number of errors found during the decompile. The error messages themselves are preceded by a ~ (tilde) and embedded in the file, at the point of error. Errors are described later in this document, under *Error Handling*.

Decompiler Summaries

At the end of each decompiled file is a summary that lists each keyword and the number of times it was decompiled, plus a total of all keywords in the file. For example, the keyword summary following a Net file lists the number of decompiled PORT keywords, NET keywords, and total keywords in the file.

In addition to its keyword summary, decompiled NC files include a device summary for each decompiled hardware object. MC keywords have a simple summary stating the number of subordinates controlled by the MC.

In addition to its keyword summary, decompiled Model files include a CSMODEL attribute summary for each decompiled CSMODEL keyword.

All summaries are commented out with asterisks.

Following are examples of an NC file keyword summary, an NC file device summary, and a Model file CSMODEL attribute summary.

NC File Keyword Summary

A keyword summary is included at the end of the NC file. This summary tells you how many objects were decompiled for each hardware and software object type, and how many features were decompiled by type. In addition, if errors occurred, ** appears after each object type with errors.

Here is an example of a summary for NC1 in the Central network. Note that the name of the network and NC appears at the top of the summary.

```
*****************************************
*             Summary for CENTRAL NC1
*
*               HARDWARE OBJECTS
*
*   D600s  1  DCDRs    0   DCMs   2  DCM140s  0
*    DSCs  1  DSC8500s 0   FIREs  0  FPUs     2
*    LCDs  0  LONs     0   N2OPENs  1
*    XMs   1**
*
*  Total Hardware Objects = 8
*
*               SOFTWARE OBJECTS
*
*    ACMs  1     ADs 22    AIs 13     AODs 13
*    AOSs  0     BDs  4    BIs 11**    BOs 13
*  C210As  0  C260As  0 C260Xs  0   C500Xs  0
*     CSs  0   DLLRs  1   JCBs 31     LCGs  0
*     MCs  0    MSDs  0   MSIs  0     MSOs 20
*   PIDLs  0  READERs 0   ZONEs 0
*
* Total Software Objects = 129
*
* Total Objects = 137 ** object counts followed by **
signify
*                        object types which had errors
*
*               FEATURES
*
*  CARDs  0  TIMEZONEs  2
*
*  Total Features = 2
```

Device Summary

A device summary follows every hardware object in a decompiled NC file, and includes information specific to the type of device (e.g., slots used, number of attached software objects).

Each hardware object type (e.g., XM, DCM, N2OPEN) has a different device summary. The data in the summary is based on the information available in the hardware object's database record.

Here is an example of an XM keyword, and the device summary that would follow it in the NC file. Note that the type of XM is indicated in the first line of the summary.

```
XM "NC2-HW", "XRL-2", "XRL IN NCU2"
    ADDRESS 6, 1, 3, 1
    GRAPHICS 0, 0
    REPORT N, N
*
* XM type is XRL
*
* Input Slots used = 1, 2, 3, 4, 5, 6, 7, 8
*
* Output Slots used = 1, 2, 3, 4, 5, 6, 7, 8
*
```

CSMODEL Attribute Summary

Each CSMODEL keyword in a Model file is followed by a summary that lists the model's attributes and their hardware references.

Here is an example of a CSMODEL attribute summary.

```
*
* attribute  hardware    attribute  hardware    attribute  hardware
* ---------  --------    ---------  --------    ---------  --------
* AI_1       AI1         AI_2       AI2         AI_3       IF1
* AI_4       IF2         AI_5       IF3         AI_6       IF4
* AI_7       AI3
*
* AO_1       IF252       AO_2       IF253       AO_3       IF254
* AO_4       IF5         AO_5       IF6         AO_6       AO1
*
* AD_1       IF7         AD_2       IF8         AD_3       ADF1
* AD_4       ADF2        AD_5       ADF3        AD_6       ADF4
* AD_7       IF9         AD_8       IF10
*
* BI_1       BI1         BI_2       BI2         BI_3       BI3
* BI_4       BI4         BI-5       BI5
*
* BO_1       BO1         BO_2       BO2         BO_3       BO3
* BO_4       BO4         BO_5       BO5         BO_6       BO6
*
* SP_1       AI4         SP_2       AI5         SP_3       AI6
* SP_4       AI7         SP_5       AI8         SP_6       IF100
* SP_7       IF101       SP_8       IF102       SP_9       IF103
* SP_10      IF104       SP_11      IF105       SP_12      IF106
* SP_13      IF107       SP_14      ADF10       SP_15      ADF11
*
```

Error Handling

The DDL decompiler keeps track of archive database errors found during the decompile, and attempts to decompile as much of the archive database as possible, despite errors.

Database errors can occur if:

- The Operator Workstation is rebooted during an upload, DDL compile, or GPL edit.
- One or more N2 objects were in an archive database downloaded to an NC which did not support the object type(s). For example, a DCM is not supported on a Fire NC. If DDL is used to create a DCM for that NC and the database is downloaded, uploaded, and then decompiled, the decompiler will produce a database error.
- Archive database files have been deleted.
- There are references to items that do not exist in the archive database, either because the file with the item cannot be found, or because the item does not exist in the file. This also occurs when an object references another object on an NC archived on a different OWS.

- A system\object name exists in the object entry database file (objent.dob) and cannot be found in the file for the specific object type. This happens when a process system\object is named but never compiled (translated).
- An object type file contains a record for a system\object that does not exist in the objent.dob file.
- An object type file contains a record for a system\object that has a different type in the objent.dob file.
- The model referenced by a CS object does not exist in the database. This happens when a CS object is added (online or DDL) to the database at one OWS and its NC is uploaded and decompiled at another OWS which does not contain the model referenced by the CS. It also occurs if a model is deleted while references to it still exist in the database.

If a database error is found during a decompile, a message explaining the error is placed in the .UND file, at the point of the error. A ~ (tilde) appears before each error message, allowing you to search quickly for the ~ to locate errors in the file.

Error messages are not commented out. Therefore, to recompile the file, you must remove error messages, or comment them out (with preceding asterisks). You must, of course, also fix the errors by making sure all references are complete and consistent in the DDL source lines, and all missing referents (e.g., a missing software model) are present.

In addition, the NC summary displays ** after each object type that had errors.

Error Examples

An example of a database error is a feedback (associated input) reference to an object that cannot be found in the NC database being decompiled. This might occur if the referenced object is in a different NC, and that NC is archived on a different PC (or all NCs were not uploaded before the decompile so that online changes are unavailable).

Here is how the message would appear in the .UND file:

```
BD "AC5", "REF-STAT"
    ASSOCINP "********", "AC3", "RF-STAT", "VALUE"
~Object doesn't exist.
    GRAPHICS 0, 0
    UNITS "OFF", "ON"
    INIT N, N, 0, 30
    REPORT N, N, , N, , 0, 0, 0
```

Note in the above example that the decompiler puts asterisks (`********`) in place of the type of object it cannot find in the database.

Another example of a database error occurs if the model referenced by a CS object cannot be found, because the Model database does not contain the referenced model.

For both these errors, the decompile continues, and an error message is placed in the .UND file, at the point of error.

NC Database Decompiling

The NC archive database decompiles are special in that the decompiler must read two files to obtain all the information about an object. First, the decompiler reads the system\object name and object type from the objent.dob file. Then, using the object type, it reads the object information from the object file. For example, if the object type were AI, the ai.dob file would be read for the object data.

Once every record in objent.dob has been decompiled, the DDL decompiler cross checks the database. Cross checking involves reading every record of every object file and comparing the system\object name and object type to the objent.dob file. If a record is found in the object type file which cannot be found in the objent.dob file, an error is generated. If the objent.dob file lists a different object type for the object, an error is generated. Cross checking the database is an extra level of error checking added to examine the NC databases for correctness.

Differences Between Decompiler and Compiler

There are some database inconsistencies that do not generate errors in the decompiler, but will generate errors if the file is recompiled.

Feedback

If a referenced feedback object (associated input) does not exist, the decompiler generates an error, and the compiler generates a warning. However, if the referenced feedback object exists but is of the wrong type, the decompiler does not generate an error or warning, but the compiler generates an error, and the compile fails.

Missing NC File

If the command line specifies a valid NC name for a network, but the NC database is not archived on the workstation, the decompile attempts to read each individual database file. A database error is generated for each read that is attempted. This allows as much of the database to be read and decompiled as possible, even in the case of a missing or corrupt file.

Recompiling

You can recompile the decompiled source code files, as long as no errors were found during the decompile, or the errors were fixed. Before recompiling, you can make modifications to the files, if necessary.

Before recompiling the source code files:

- Make sure all database error and warning conditions are fixed. Make sure all error messages (preceded by ~) are removed or commented out.
- Rename the .UND files to .DDL.
- Remember that process objects do not remain intact when you compile the NC source code files. Therefore, along with a recompile of an NC file, you must recompile all process objects at the workstation (or retranslate with GPL).

Compatibility With Previous Release Databases

The DDL decompiler can decompile Release 6.0, 7.0, and 8.0 archive databases. Databases that have been created and uploaded at previous Metasys software releases must be converted before they can be decompiled. For the decompiler to execute properly, the most current revision of DDL (which includes the decompiler) must be installed on the workstation performing the decompile.

When decompiling a previous release database, consider the following:

- Release 1, 2, and 3.00 databases uploaded from an NCM may have garbage in the XRL and XRM slot tables, beyond Slot 8. Ignore this information. This has been fixed for release 3.01 and subsequent releases.
- Databases uploaded from a Release 2 NCM may have garbage in the N2Open slot tables. Ignore this information. At Release 2, mapping into these slots was not allowed. At Release 3 and beyond, mapping into these slots is allowed and garbage is not uploaded.
- When the decompiler is cross checking an NC database (as described in this section, under *NC Database Decompiling*), and it attempts to cross-check an object type which did not exist at a previous release (e.g., Zone did not exist at Release 2), it generates an Open Database File error. Decompiling continues, and an error message is placed in the .UND file.
- Databases uploaded from Release 1, 2, or 3 will decompile with two error messages, one for each of the two Access feature databases (TIMEZONE and CARD) supported by DDL. (These are Release 4 features that did not exist in previous releases.) Decompiling continues, and an error message is placed in the .UND file.
- Databases uploaded from Release 1, 2, 3 4, 5.00, or 5.01 will decompile with an error message for the DCM140 database, which was added at Release 5.02. Decompiling continues, and an error message is placed in the .UND file.
- Databases uploaded from Release 1, 2, 3, 4, 5, or 6.0 will decompile with an error message for the MC database, which was added at Release 7.00. Decompiling continues, and an error message is placed in the .UND file.

- For Release 8.0, the DLLR Projection Time attribute has been replaced by the Sensitivity attribute. If an earlier database is decompiled using the Release 8.0 decompiler, the Projection Time attribute will be converted to the Sensitivity attribute with a default value of Medium. A message at the end of the .UND file will indicate this conversion if it occurs.
- Databases uploaded from Releases 1 through 9 will decompile with an error message for the LON database which was added at Release 10. Decompiling continues and an error message is placed in the .UND file.

Advanced Topics

Advanced Topics includes:

- answers to some application questions
- explanations of some error and warning messages

Application Questions

This section provides answers to some application questions concerning DDL use.

Changing an NCM Node or Subnet Address

How do I change the node or subnet address of one NCM on the network?

1. Upload the NCM whose node address you want to change.
2. Upload all NCMs that have references to the NCM whose address you are changing. (References are AD, BD, MSD, MC associated objects, GPL and JCB references, Demand Limiting/Load Rolling loads, etc.) It is safest to upload every NCM.
3. Upload Global database.
4. Decompile NCM whose address you are changing.
5. Decompile Global database.
6. Edit decompiled Global file to change NCM node or subnet address.
7. Rename edited Global file from .UND to .DDL extension. Recompile Global file.

8. Rename NC file created in Step 4 from .UND to .DDL extension. Recompile NC file.
9. Retranslate all GPL for NCM, if any.

   Recompile all JC-BASIC for NCM, if any.
10. Change NCM's node or subnet address with NC Setup.
11. Download Global database.
12. Download NCM whose address has changed.
13. Download all NCMs that have references to the changed NCM. Any NCM not downloaded may be unable to reference objects in the changed NCM.

Moving an NCM to a Different Network (Network Name Changes)

How do I move an NCM from one network to another without having to re-add all feature data?

Notes: Use this procedure if the network name changes, but the NCM name does not.

Steps 9, 10, 13 (all marked with *****) **are optional** if the NCM's node address is not changed as a result of the move.

This procedure assumes the PC used for the DDL compile and decompile has access to both networks, and the user knows how to properly switch between the two to perform the necessary operations.

In the following procedure, FMSData refers to the Metasys profile string in the WIN.INI file.

1. Upload the NCM you want to move.
2. Decompile the NCM
3. Edit decompiled NC file to change network name on @NC line. Edit any references to objects not on the new network.
4. Upload the Global database of the network you are moving the NCM to.
5. Decompile the Global database.
6. Add the NCM to the network to which you want to move it. To add the NCM, you have three options: 1) edit the DDL Global file and recompile; 2) create a new incremental Global file with the NCM and its systems, and do an incremental compile of the file; or 3) use online generation and perform a global upload.

7. Move the entire directory (and subdirectories) from:

   FMSData\old network\DEVICES\NCM name

   to:

   FMSData\new network\DEVICES\NCM name

8. Move the entire directory (and subdirectories) from:

   FMSData\old network\GPL\NCM name

   to:

   FMSData\new network\GPL\NCM name

   Note: At Release 5 and beyond, subdirectories of FMSData\network\GPL are no longer created automatically. This means you must move all GPL files for the NCM being moved, using whatever directory scheme desired.

   Remember that Steps 9, 10, 13 (all marked with *) are optional if the NCM's node address is not changed as a result of the move.

*9. Recompile edited NC file with new network name.

*10. Retranslate all GPL for NCM, if any.

   Recompile all JC-BASIC for NCM, if any.

11. Change the NCM's network name with NC Setup.

12. Connect NCM to network.

*13. Change NCM's address with NC Setup.

14. Download Global database.

15. Download NCM.

16. If the old network continues to exist, delete the NCM and all its systems from the old network. Also, delete all references to the deleted NCM.

Changing an NCM Name

How can I rename an NCM on a running network?

Note: The following procedure will work if the **only** change to the Global file is the NCM's name.

In the following procedure, FMSData refers to the Metasys profile string in the WIN.INI file.

1. Upload the NCM to be renamed.
2. Decompile the Global database.
3. Edit decompiled Global file to change NCM name. (Edit the NC keyword, NCM parameter, and the SYS keyword, NC owner name parameter.)
4. Change the directory name from:

   FMSData\network\DEVICES\old NCM name

   to:

   FMSData\network\DEVICES\new NCM name
5. Note: The GPL directory structure changed at Release 5. If there is a user-defined structure, you will need to decide if any action is necessary.

   Change the directory name from:

   FMSData\network\GPL\old NCM name

   to:

   FMSData\network\GPL\new NCM name
6. Compile edited Global file.
7. Edit GPL (or JC-BASIC) files for new NCM name.
8. Retranslate all GPL (or recompile all JC-BASIC) processes.
9. Download Global database.
10. Download NCM.

Changing a Network Name

What is the easiest way to change my network name on a job that is running?

For this discussion, let's assume the existing network name is OLD, and the new network name is FUTURE. Also, in the following procedure, FMSData refers to the Metasys profile string in the WIN.INI file.

1. On every OWS on the network:
    a. Get a printout of a current NET DDL file. (Decompile the NET file if a current one is not available.)
    b. Edit the NET file to change the network name from OLD to FUTURE.
    c. Rename FMSData\OLD directory to FMSData\FUTURE.
    d. Rename FMSData\OLD.dob directory to FMSData\FUTURE.dob.
    e. Compile the NET file that has been changed.
2. Run NC Setup on every NC on the network to change the network name.
3. Do a SETFMS command to restart the Metasys system, and do a Global Download on every OWS on the network.

Recompiling NC Files After Compiling Global or Model Files

Why do I have to recompile my NC files after compiling my Global or Model file?

Generating a database is a building process. Each DDL compile builds on the data in the previous file. (That is why there is a proper order for compilation of files: Net, Model, Global, NC.)

For example, when an NC is defined in the Global file, its node address is specified. The NC node address is stored in the database and when the NC compile occurs, this node address is added to the database record for every hardware object (e.g., DCM, ASC) and every software object (e.g., AI, BO) generated by the NC compile. A full compile of a Global or Model file changes values in the database that the NC may use.

Similarly, a full compile of a NET file requires a recompile of all Global and NC files.

Retranslating GPL Files after Compiling NC Files

Why do I have to retranslate my GPL files (recompile my JC-Basic files) after compiling my NC files?

The GPL/JCB processes are the least independent part of the database building process. Information about the objects read, written, and commanded by processes are stored in the process database. All changes made by the NC compile to objects referenced in a process need to be incorporated into the process database. The DDL NC compile removes all archive database links to GPL/JCB process objects on the compiled NC. DDL puts only the process name into the object entry database file for every JCB keyword in the DDL NC source file. In order to download process objects with the newly compiled NC database, the GPL files must be retranslated (JC-Basic files must be recompiled).

If the DDL NC compile created ALL the objects that a GPL file references (for all object blocks), then it is only necessary to open the file and start the translation. However, if the GPL file references objects no longer in the archive database, it will be necessary to do a “session read and F10” for any object not in the archive database.

NET File for OWS with Multiple Networks

What does a NET file look like for an OWS with multiple networks?

Here is an example of a DDL NET file for an OWS with multiple networks. Guidelines follow the example.

```
@NET
*
*     port name
*     |     port use (3=printer, 1=NCDial)
*     |     | baud rate
*     |     | |
PORT "LPT1",3
PORT "COM1",1,9600
*
*    network name
*    |          description
*    |          |                   PC Name
*    |          |                   |
NET "JCITOWER","JCI Headquarters","PC1"
*
*            PC-Subnet address
*            | PC-Node address
*            | |
    N1DIRECT 1,100
*
*     port name
*     |     port use (2=NCDirect)
*     |     | baud rate
*     |     | |
PORT "COM3",2,9600
NET "A-BLDG", “A BUILDING, NC-DIRECT",
    NCDIRECT
*
*
*
NET "Z-BLDG", “Z BUILDING, NC-DIAL", "PC1"
*
*           dial type (tone or pulse)
*           |    phone number
*           |    |
    NCDIAL "T", "414-555-1234, #5039"
```

Keep the following in mind for multiple network NET files.

- For NCDial connections, a PORT keyword that defines the port as NC-Dial must come before the NET keyword. The above example defines COM1 as NCDial for the Z-BLDG network.

- For NC-Direct connections, a PORT keyword that defines the port as NC-Direct must come before the NET keyword. The above example defines COM3 as NCDirect for the A-BLDG network.
- If the OWS name is specified on the NET keyword line, that name must be defined as a PC in the Global file for the network. Therefore, the above example requires definition of PC1 in the Global files for the "JCITOWER" and "Z-BLDG" networks.
- You will need one Global file per network. (Therefore, the above example requires three Global files).

Configuring DDL Files for the M5 Workstation

Where can I find information on configuring DDL files for the M5 Workstation?

The multiple N1 Configuration process is described in the *N1 Ethernet/IP Network Technical Bulletin (LIT-6360175)*. This document is located in the *Metasys Network Technical Manual (FAN 636)*.

NET File Keyword Order

Why does a NET file that compiled fine at Release 1, 2, 3, or 4 no longer compile?

At Release 5, additional error checking was added to the DDL compiler. The additional error checking requires that PORT definition precede NCDIRECT and NCDIAL network definition. Simply stated, the PORT definition must precede the NET keyword for NCDIRECT and NCDIAL networks.

As seen in the example on the preceding page (under the heading *NET File for OWS with Multiple Networks*, the PORT keywords can be placed at the beginning of the file (this is where UNDDL puts them). Or they can be interspersed with the NET keywords, as long as at least one PORT keyword defining a NCDial port precedes all NCDIAL networks, and at least one PORT keyword defining a NCDirect port precedes all NCDIRECT networks.

What is Going on During a DDL Compile

Why, during a compile, is the disk being hit three times for each dot in the NC file?

All DDL compiles are disk intensive; each compile type builds a portion of the database on the disk. The DDL compiler must access a disk file for every message it puts on the screen or in the list file. All compile types go through the same initial steps, and each one causes one or more disk accesses:

- opening the source file
- printing copyright information to the screen
- opening the list file and printing copyright information to the file
- creating a backup of the portion of the database to be compiled

All compile types follow the same procedure for generating the database. Each line of the source file is read in, processed, and written to the list file. Every read or write is a potential disk access. The number of times a disk is "hit" depends on:

- if a disk cache is installed and which one
- the position of the files on the disk
- the number of disk accesses required by the keyword for semantic rules and adds to various databases

Compiling each keyword involves making all the semantic checks and, if there are no errors, adding the record to the database. The semantic checks may involve reading from the database more than once. In addition, adding the record may involve writing to more than one database file.

For example, each BI keyword involves one database read to find the system name and verify that it is on the same NC as the BI, one database read to verify that the object name does not already exist, and one database read to verify that the hardware (let's assume XM) exists, is on the same NC, and its specified slots are free. To add this BI record to the database, its name and type must be written to the object entry database, its data must be written to the BI database, and its slot use must be written to the XM slot table. After the add of this BI, a dot is written to the screen. If an error is found in the BI, the add will not take place; instead, an error message, read from a disk file, is written to the list file, and an asterisk is written to the screen.

Error and Warning Messages

This section provides explanations of some of the error and warning messages that can occur during a compile. These messages are printed in the compiler list file (.DLT) described in the *Compiler* section of this manual.

The messages as seen on the screen or in the list file are listed in alphabetical order.

Compile End Failed

Check that none of the files in your database are write protected. Also, delete any of the following files, if they exist. (In the file names, *Data* and *FMSData* refer to the environment variables in the WIN.INI file.)

*Data**Data*.tmp

FMSData\[network].tmp

FMSData\[network]\devices\[ncname].tmp

Compile Start Failed

Check that there is sufficient disk space for both a copy of the database being compiled and any new database files.

DBM Init Failed

Check the path to which the environment variable FMSDOS points. There must be a WIN.INI file properly set up in this directory. To find out where FMSDOS currently points, type SET at the DOS prompt and press Enter.

For information on proper setup of your WIN.INI file, see the *Compiler* section of this manual.

Using the Win Prep-For utility is one way to make sure the FMSDOS variable points to the correct directory.
Win Prep-For is described in the *Win Prep-For Utility Technical Bulletin (LIT-1201656)*.

Note: If you are using the MS-DOS version of PREP-FOR see the *Move Utility Technical Bulletin (LIT-636110)*. **Do not** use the MS-DOS version of PREP-FOR with M5 Workstations.

?Warning: Software Reference Table full "sys\obj" must be checked manually for existence and proper type.

The Software Reference Table is a list of non-existent objects that are referenced by other objects. The compiler adds the non-existent object to the table when the reference to the non-existent object occurs. (Perhaps the object does not yet exist, because it will be added later in the compile.)

Examples of software references to other objects are an AD or BD's associated object, a BO or AOS's feedback object, a PIDL's input or output object, a DLLR's meter, and a BO load's DLLR object.

The table can hold a maximum of 200 entries. If the Software Reference Table is full, the above warning is placed in the .DLT file at the point of reference to the non-existent object. In this case, it is the user's responsibility to manually check that the object does actually exist and is properly referenced.

When an NC compile (full or incremental) successfully finishes Pass 1, the compiler examines the Software Reference Table one entry at a time to determine if the referenced object exists. If the object still does not exist, a warning is placed in the .DLT file. If the object does exist (having been added later in the compile), error checking verifies that the object is referenced properly.

To ensure that the table does not become full, therefore allowing the compiler to check for proper referencing (rather than the user), order the objects in the DDL source file so that referenced objects appear before the objects that reference them.

? Warning: “sys\obj” does not exist, referenced by “sys2\obj2”

When a software object is compiled, during an NC compile, that references another object which does not exist in the archive database, an entry is made into the Software Reference Table. When the NC compile (full or incremental) successfully finishes compiling all objects, the compiler examines the Software Reference Table, one entry at a time, to determine if the referenced object exists. If the object still does not exist, this warning is placed in the .DLT file. If the object does exist (having been added later in the compile), error checking verifies that the object is referenced properly.

Examples of software references to other objects are an AD, BD, or MC associated object, a BO or AOS feedback object, a PIDL input or output object, a DLLR meter, an MC subordinate object, and a BO load DLLR object.

If the referenced object is on the same NC being compiled, change the situation, because the object doing the referencing will not function properly. For example, an AD associated to a non-existent AI will always be set to its initial value.

If the referenced object is on a different NC, there are many reasons why the object may not exist in the archive database. For example, the OWS where the DDL NC file is compiled may not be the archive for that NC or that NC may not have been compiled yet.

If the referenced object does not exist on the network when this newly compiled NCM is downloaded, runtime errors may occur.

ERRORLOG.TXT File Messages

The following message appears in the ERRORLOG.TXT file.

`Memory Allocation Failure.`

If there is not enough memory to run the compiler, a number appears on the screen after the last `.` or `*`, instead of the typical:

```
DDL Compile complete
Errors:    None
Warnings:  None
```

In this situation, a message also appears in the ERRORLOG.TXT file about Memory Allocation (e.g., Global, Local). This indicates there is not enough conventional memory to run the DDL compiler. Specifically, there is not enough memory to open the DDL language file.

This usually occurs when the DDL compiler is executed from a "DOS window." If it happens when Windows software is not running, then another program (e.g., DOS shell, mouse driver) must be removed to provide enough memory for DDL to compile.