800xA for MOD 300

Configuration

System Version 6.0

Power and productivity
for a better world™
800xA for MOD 300

Configuration

System Version 6.0
NOTICE

This document contains information about one or more ABB products and may include a description of or a reference to one or more standards that may be generally relevant to the ABB products. The presence of any such description of a standard or reference to a standard is not a representation that all of the ABB products referenced in this document support all of the features of the described or referenced standard. In order to determine the specific features supported by a particular ABB product, the reader should consult the product specifications for the particular ABB product.

ABB may have one or more patents or pending patent applications protecting the intellectual property in the ABB products described in this document.

The information in this document is subject to change without notice and should not be construed as a commitment by ABB. ABB assumes no responsibility for any errors that may appear in this document.

Products described or referenced in this document are designed to be connected, and to communicate information and data via a secure network. It is the sole responsibility of the system/product owner to provide and continuously ensure a secure connection between the product and the system network and/or any other networks that may be connected.

The system/product owners must establish and maintain appropriate measures, including, but not limited to, the installation of firewalls, application of authentication measures, encryption of data, installation of antivirus programs, and so on, to protect the system, its products and networks, against security breaches, unauthorized access, interference, intrusion, leakage, and/or theft of data or information.

ABB verifies the function of released products and updates. However system/product owners are ultimately responsible to ensure that any system update (including but not limited to code changes, configuration file changes, third-party software updates or patches, hardware change out, and so on) is compatible with the security measures implemented. The system/product owners must verify that the system and associated products function as expected in the environment they are deployed.

In no event shall ABB be liable for direct, indirect, special, incidental or consequential damages of any nature or kind arising from the use of this document, nor shall ABB be liable for incidental or consequential damages arising from use of any software or hardware described in this document.

This document and parts thereof must not be reproduced or copied without written permission from ABB, and the contents thereof must not be imparted to a third party nor used for any unauthorized purpose.

The software or hardware described in this document is furnished under a license and may be used, copied, or disclosed only in accordance with the terms of such license. This product meets the requirements specified in EMC Directive 2004/108/EC and in Low Voltage Directive 2006/95/EC.

TRADEMARKS

All rights to copyrights, registered trademarks, and trademarks reside with their respective owners.

Copyright © 2003-2016 by ABB.
All rights reserved.

Release: September 2016
Document number: 3BUR002417-600 B
## Table of Contents

### About This User Manual

- User Manual Conventions .................................................................................................................. 12
- Warning, Caution, Information, and Tip Icons ...................................................................................... 12
- Terminology ........................................................................................................................................ 13
- Released User Manuals and Release Notes ......................................................................................... 14

### Section 1 - Overview

- Introduction ........................................................................................................................................ 15
- Configuration Guidelines ...................................................................................................................... 16

### Section 2 - Administration

- Introduction ........................................................................................................................................ 17
- Before You Get Started ......................................................................................................................... 17
  - Database Requirements for Systems with MOD 300 Software ......................................................... 18
  - Time Synchronization ......................................................................................................................... 18
- Users .................................................................................................................................................. 19
- Add Server and Client Nodes .............................................................................................................. 19
- Set Up System Structure ..................................................................................................................... 19
  - Enable the MOD OPC Service Groups ............................................................................................ 25
  - Enable the MOD 300 Batch OPC Service Groups .......................................................................... 26
  - Complete the Service Structure Set Up ............................................................................................. 26

### Section 3 - MOD Importer

- Introduction ........................................................................................................................................ 29
  - Loop Type Change ............................................................................................................................. 31
- Using MOD Importer ............................................................................................................................ 31
Table of Contents

Application Interface ........................................................................................... 31
User Interface ...................................................................................................... 32
Application Start-up and Exit .............................................................................. 32
Navigator Display ............................................................................................... 33
Import MOD Objects and Data ............................................................................ 35
Find MOD Objects ............................................................................................... 37
  Using Find ........................................................................................................ 37
  Browse Form .................................................................................................... 39
  Show Ancestors ............................................................................................... 39
View MOD Importer Log File ............................................................................... 40
Setting MOD Importer Options .......................................................................... 41

Section 4 - MOD Environment Importer
Introduction .......................................................................................................... 45
Using MOD Environment Importer ..................................................................... 45
  Application Start-up and Exit ......................................................................... 46
  User Interface .................................................................................................. 47
  Navigator .......................................................................................................... 48
Setting Options .................................................................................................... 50
Import MOD Environment Objects and Data ..................................................... 51
MOD Object Tag Status ....................................................................................... 51
View Log File ....................................................................................................... 53

Section 5 - MOD 300 TCL Phase Importer
Introduction .......................................................................................................... 55
Using MOD TCL Phase Importer ....................................................................... 55
  Application Start-up and Exit ......................................................................... 56
  User Interface .................................................................................................. 57
Setting Options .................................................................................................... 58
Import MOD Phase Objects ................................................................................ 59
View Log File ....................................................................................................... 60

Section 6 - Configuration
# Table of Contents

MOD OPC Server Configuration ................................................................. 62  
Fixed Rate MOD 300 OPC Server Configuration ........................................... 63  
Composite (_COMP) Configuration for MOD Trend Data Quality and History Data 63  
Define the MOD OPC Data Source .............................................................. 65  
Assigning Multiple Connectivity Servers .................................................... 66  
Alarm Routing with Multiple Connectivity Servers ....................................... 66  
Running MOD 300 Utility Aspect Report .................................................... 67  
Running MOD OPC Statistics Aspect ............................................................ 68  
Alarm Event Configuration ........................................................................... 69  
Data Access Configuration .......................................................................... 69  
MOD Alarm and Event Logger Configuration ................................................ 69  
Audit Trail and Parameter Change Messages ............................................... 70  
Control Structure Configuration ................................................................... 71  
Control Structure Configuration Using MOD Templates ............................... 71  
MOD_Sequence Object ................................................................................ 76  
Special Permissions for MOD Objects .......................................................... 76  
Creating a MOD CCF Loop Object ............................................................... 81  
Environment Structure Configuration ......................................................... 82  
Configuring a MOD Environment ............................................................... 83  
Configuring a MOD Environment Area Object ............................................ 83  
Configuring a MOD Environment Group Object ......................................... 84  
Graphics ......................................................................................................... 89  
Dynamic Value .............................................................................................. 90  
Add a Graphic Aspect to an Object and Open Graphic Builder ...................... 90  
Using MOD_DValue in Graphic Builder ....................................................... 91  
Using Pre-configured DValues in Graphic Builder ........................................ 92  
MOD_DValue Format Definition for Process Graphics 2 .............................. 92  
TCL Array Plot ............................................................................................... 95  
Create MOD 300 Plot Aspect ....................................................................... 95  
Configure MOD 300 Plot Aspect ................................................................. 96  
MOD 300 Group Trend Template ................................................................. 104
MOD 300 Trend Buffer................................................................. 105
MOD 300 Alarm Colors Aspect .................................................. 106
MOD 300 Display Colors .......................................................... 107
MOD 300 NLS Resources .......................................................... 108

Appendix A - Advant OCS Database
GENERICD Object ................................................................. 113
   PHYSICAL DEVICE .............................................................. 114
   AUTO START ........................................................................ 115
   SOFTWARE NAME ............................................................. 115
   DO RATE ............................................................................. 115
   SECONDARY DP, BACKUP ENABLE, and BACKUP OVERRIDE .. 115
   DEFAULT ENVIRONMENT .................................................... 115
   NODE TYPE ........................................................................ 115
AREA Object ................................................................................ 116
   Message Center Fields .......................................................... 117
   Message Center Entries for Loggers ......................................... 118
   Message Routing from Operator Station .................................. 118
   Recommended Message Centers for Routing .......................... 119
Changes to Loop Type ................................................................ 119
Memory Pool Utility (mputil) ..................................................... 119

Appendix B - Syntax for Database References
   Loop Parameter Reference (Tag Access) Format ...................... 124
   FCM Parameter Reference (Tag-FCM Access) Format .............. 125
   Array Element Parameter Reference Format ........................... 126
   Recipe Item Parameter Reference Format ............................... 127
   Taylor Ladder Logic (TLL) Parameters Reference Format .......... 128

Appendix C - Batch Support
Introduction ............................................................................... 131

Appendix D - Generic Tag Configuration
Table of Contents

Digital Loop Properties Reference ................................................................. 144
Analog 1 (Continuous) Loop Properties Reference ........................................... 147
Analog 2 (Control) Loop Properties Reference .................................................. 149
Analog 3 (PID) Loop Properties Reference ....................................................... 151

Appendix E - TCL Batch Commands and Configuration to Support PDL
Introduction ........................................................................................................ 155
TCL Batch Commands .......................................................................................... 155
  TCL Batch Command Configuration - REPORT ............................................. 156
Configuration to Support PDL ............................................................................. 157
  Report Builder (REP_BLDR) Database Object ............................................ 157
  Report ID Fields in Unit Master Database Objects ......................................... 157

Appendix F - Customized Data for Backup
Introduction .......................................................................................................... 159
PAS Software Communication Settings ............................................................ 159
Operational Registry Settings ............................................................................. 160
Fixed Registry Settings ....................................................................................... 164
Customizing Objects ........................................................................................... 165
Custom Color Definitions .................................................................................. 165

Revision History
Introduction .......................................................................................................... 167
Revision History .................................................................................................. 167
Updates in Revision Index A .............................................................................. 167
Updates in Revision Index B ............................................................................... 168

Index
About This User Manual

Any security measures described in this User Manual, for example, for user access, password security, network security, firewalls, virus protection, etc., represent possible steps that a user of an 800xA System may want to consider based on a risk assessment for a particular application and installation. This risk assessment, as well as the proper implementation, configuration, installation, operation, administration, and maintenance of all relevant security related equipment, software, and procedures, are the responsibility of the user of the 800xA System.

This user manual describes how to configure 800xA for MOD 300. This user manual covers procedures for:

- Configuring 800xA with MOD 300.
- Importing MOD tag database.
- Importing MOD environment structure.
- Adding additional tags.
- Building graphics using MOD submodels.
- Additional database configuration requirements.
- Assigning access rights to objects.

As a prerequisite you should be familiar with the 800xA System, MOD 300, and your AdvaBuild project for the Advant OCS with MOD 300 control system. In addition, you must have Administrator privileges on the computer where the software runs.
User Manual Conventions

Microsoft Windows conventions are normally used for the standard presentation of material when entering text, key sequences, prompts, messages, menu items, screen elements, etc.

Warning, Caution, Information, and Tip Icons

This User Manual includes Warning, Caution, and Information where appropriate to point out safety related or other important information. It also includes Tip to point out useful hints to the reader. The corresponding symbols should be interpreted as follows:

- **Electrical warning icon** indicates the presence of a hazard that could result in *electrical shock*.

- **Warning icon** indicates the presence of a hazard that could result in *personal injury*.

- **Caution icon** indicates important information or warning related to the concept discussed in the text. It might indicate the presence of a hazard that could result in *corruption of software or damage to equipment/property*.

- **Information icon** alerts the reader to pertinent facts and conditions.

- **Tip icon** indicates advice on, for example, how to design your project or how to use a certain function.

Although Warning hazards are related to personal injury, and Caution hazards are associated with equipment or property damage, it should be understood that operation of damaged equipment could, under certain operational conditions, result in degraded process performance leading to personal injury or death. Therefore, fully comply with all Warning and Caution notices.
A complete and comprehensive list of terms is included in *System 800xA System Guide Functional Description (3BSE038018*)*. The listing includes terms and definitions that apply to the 800xA System where the usage is different from commonly accepted industry standard definitions and definitions given in standard dictionaries such as Webster’s Dictionary of Computer Terms. Terms that uniquely apply to this User Manual are listed in the following table:

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC410, AC460</td>
<td>Advant Controller 410 and 460</td>
</tr>
<tr>
<td>AdvaBuild</td>
<td>Configuration package for Advant OCS running on an Engineering Station (HP-UX or Windows)</td>
</tr>
<tr>
<td>Advant OCS</td>
<td>ABB Advant Open Control System</td>
</tr>
<tr>
<td>CCF</td>
<td>Configurable Control Functions</td>
</tr>
<tr>
<td>CDP</td>
<td>Configurator Data Processor. A configuration node on the MOD 30 System like the Advant Engineering Station with AdvaBuild.</td>
</tr>
<tr>
<td>DCN</td>
<td>Distributed Communication Network</td>
</tr>
<tr>
<td>Environment</td>
<td>An environment is a portion of the database that you can view using group and area displays. Several environments can be defined for a system.</td>
</tr>
<tr>
<td>FCM</td>
<td>MOD 300 Function Class Module. FCMs are configured to form loops.</td>
</tr>
<tr>
<td>mputil</td>
<td>Memory Pool Utility</td>
</tr>
<tr>
<td>Node level objects</td>
<td>CNTRLLER, CONSOLE, GENERICD, AC410, AC460MOD, BUC, DCN_DCN, ADVANT_D2D</td>
</tr>
<tr>
<td>TCL</td>
<td>Taylor Control Language (MOD sequence control)</td>
</tr>
<tr>
<td>TLL</td>
<td>Taylor Ladder Logic</td>
</tr>
</tbody>
</table>
Released User Manuals and Release Notes

A complete list of all User Manuals and Release Notes applicable to System 800xA is provided in System 800xA Released User Documents (3BUA000263*).

System 800xA Released User Documents (3BUA000263*) is updated each time a document is updated or a new document is released. It is in pdf format and is provided in the following ways:

- Included on the documentation media provided with the system and published to ABB SolutionsBank when released as part of a major or minor release, Service Pack, Feature Pack, or System Revision.
- Published to ABB SolutionsBank when a User Manual or Release Note is updated in between any of the release cycles listed in the first bullet.

A product bulletin is published each time System 800xA Released User Documents (3BUA000263*) is updated and published to ABB SolutionsBank.
Section 1 Overview

Introduction

The following is a quick overview to the sections in this user manual.

Where to Start

Section 2, Administration describes preparations necessary to configure MOD 300 for the 800xA System using the Configuration Wizard.

How to Import

Refer to Section 3, MOD Importer for instructions on importing MOD tag object configuration data from an AdvaBuild project and populating the Aspect Server.

A similar importer is used to import MOD Environment Area and MOD Environment Group object as described in Section 4, MOD Environment Importer.

See Section 5, MOD 300 TCL Phase Importer when information from the MOD Engineering Station TCL Server library is needed to populate the Control Structure in the 800xA system with MOD Phase objects. MOD_UNIT objects must already be imported.

Configuration Activities

Refer to Section 6, Configuration for instructions on setting up tags and graphics for the MOD 300 user interface from within the Plant Explorer. The operator environment (the set of displays, authorities, and operations that a user has access to) is described here. In addition, any special conversions or configuration options are included.
Configuration Guidelines

- **MOD subelements/graphic symbols**
  The following aspects are supplied and maintained by ABB, they should not be updated or modified. Any changes made to these symbols will be lost when upgrading to a newer version of 800xA for MOD 300. If changes are needed they need to be documented by the customer and reapplied after upgrading.
  - Faceplates.
  - MOD_DValue.
  - Measure_DValue, etc.

- **Control Structure**
  The control structure must match the structure in AdvaBuild and can only contain MOD object types under the root level object MOD_DB. Additional objects or objects not in the proper location may cause problems with view/retrieving data.
  MOD 300 objects are matched by IDs to the imported Control Structure. If the Control Structure is deleted, then the same must be done for the object. Otherwise, the object will not match with the Control Structure when it is imported again (object must match with the control structure to work).

- **MOD Object Types**
  All MOD objects must be based on the MOD object types. These object types may not be subclassed or superclassed.

- **MOD Environment structure**
  The hierarchy of MOD_ENV -> MOD_ENVAREA -> MOD_ENVGROUP -> MOD_ENVBLOCK -> object must be maintained in this structure. Additional objects of non MOD_ENV_ * types may cause problems with viewing/retrieving data.

- **MOD Configuration Aspects**
  Templates, Colors, and Alarm pages aspects may be changed/updated but all changes should be documented in the event that these items are modified in a subsequent product release.
Section 2  Administration

Introduction

This section describes preparations necessary to configure the 800xA for MOD 300 system. This section also serves as a guide to relevant 800xA system administration activities. References to other documents are made to support setting up MOD 300.

If the automated installer is used to set up the system, then the information on creating a system, adding system extensions, adding server and client nodes, and setting up the system structure can be skipped. Service groups can be enabled after the importers are run.

Before You Get Started

The procedures in this section assume any of the following configurations already installed:

- Connectivity Server with Client.
- Aspect Server with Client.
- Client.

An Aspect Server is required to import configuration data using the MOD Importer. In addition, an AdvaBuild for Windows engineering station node must also be present.

This user guide includes a number of references to the Real-Time Accelerator Board (RTAB). These references and comments also apply to the PU410 and PU412 RTA units.

See the system book for operator workplace configuration before going into the details in this user manual. The Engineering of the controllers is described in the AdvaBuild User’s Guides.
Database Requirements for Systems with MOD 300 Software

The Connectivity Server with MOD 300 and the Real-Time Accelerator Board (RTAB) connects to the Distributed Communications Network (DCN) and requires a GENERICD object to establish the server as a node in the MOD 300 database.

The GENERICD database object is inserted into the MOD 300 database hierarchy and its object attributes are defined via the AdvaBuild Structure Builder and Template Builder respectively as described in the AdvaBuild Basic Functions User’s Guide or in the AdvaBuild Control Builder for Windows User’s Guide. See Appendix A, Advant OCS Database for guidelines.

Time Synchronization

Time Synchronization on the Windows side of a DCN node is set by default so Reverse Time Synchronization (RTS) is disabled. This allows the host Connectivity Server node to constantly be updated by the Real-time Accelerator Board (RTAB) clock which gets its time from the MOD System Master Timekeeper. The Connectivity Server node then broadcasts this time to the other 800xA for MOD 300 nodes.

The following rules apply when RTS is disabled:

- The RTAB time is synchronized with the master timekeeper node
- Host clocks are set by time synchronization messages from their RTAB clock
- The node cannot become master via a console time entry when an RTS node is master timekeeper. The operator time entry is ignored with the diagnostic:

  OPERATOR TIME CHANGE REQUEST IGNORED

The RTS mode is set in the host configuration file via a diagnostic logged in the diagnostic archive at node backup:

SSE: REVERSE TIME SYNCH ENABLED
SSE: REVERSE TIME SYNCH DISABLED
Users

The 800xA System uses Windows administration rights as the basis for its security system. There are five predefined users groups in the 800xA System (the last three groups have specific MOD 300 object permissions). The user rights associated with your logon name and password determine your security level.

<table>
<thead>
<tr>
<th>User Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Everyone</td>
<td>A group of all users that allows read permission.</td>
</tr>
<tr>
<td>Administrators</td>
<td>A group with the security system disabled. Members of this group have full access to everything in the aspect system.</td>
</tr>
<tr>
<td>Operators</td>
<td>Operators group operates the process.</td>
</tr>
<tr>
<td>Application Engineer</td>
<td>Application engineers group performs application engineering such as: configure, tune, enter, download.</td>
</tr>
<tr>
<td>System Engineers</td>
<td>System engineers group performs system engineering such as: shutdown, security configure, administrate, supervise.</td>
</tr>
</tbody>
</table>

See *System 800xA Administration and Security (3BSE037410*)*, for a complete description of user security. The user account that installed the 800xA system is added to the system as member of Everyone, System Engineer, Application Engineer, and Administrator. Log into the server station as an Administrator to perform the system configuration work described in this section.

Add Server and Client Nodes

Refer to *800xA Installation, Update and Upgrade Getting Started manual* located at C:\Program Files (x86)\Common Files\ABB Industrial IT\800xA\System\Documentation\System 800xA. This can be accessed from the ABB Start Menu > Documentation.

Set Up System Structure

Create the following system structures before importing. To prepare the system before using the MOD Importer, follow these steps.
1. From the Control Structure of the Plant Explorer Workplace, use the context menu of the Root Domain and select **New Object**, Figure 1. Additional information on object configuration is included in the system levels.

![Figure 1. Select New Object](image)

2. Select MOD OPC Server Network from the New Object display as shown in Figure 2 and then enter a name for the object. For the MOD 300 Batch Server, use MOD OPC Batch Server Network instead. Click **Next** to continue. Use a name that represents an area or group of connectivity servers not a specific one.
Figure 2. Setup MOD OPC Server Network Object
3. Use **Additional Arguments** to **Add...** one or two Connectivity Server(s). Select the primary Connectivity Server first. Click **OK**, **Figure 3**, when done.

![Select Connectivity Server(s)](image)

**Figure 3. Select Connectivity Server(s)**

4. Once the Connectivity Server is added, the Selected OPC Server ProgID is displayed in the window.
   
   For MOD 300 this will be:
   
   AdvOPCDAServer.OPCDAServer.1.
   
   For MOD 300 Batch this will be:
   
   AdvMODBatchOPCDAServer.MODBatchOPCDAServer.1.
   
   If this is correct, click **Create** to create the Service Group.
Within the Control Structure, the MOD OPC Server Network object is created. Also, within the Service Structure, the MOD OPC DA and MOD OPC AE service groups are created.

Repeat the above steps for additional Connectivity Servers.

5. Open the Service Structure to see the MOD OPC DA (OpcDA_Connector, Service) and MOD OPC AE (EventCollector, Service) service groups, Figure 4. By default, these service providers are not enabled. Only enable them after running the MOD Importer.

Neither the MOD 300 OPC DA server nor the MOD 300 Batch OPC DA server supports parallel redundancy. The Allow Parallel Redundancy check box must be unchecked.

The parameter is located in the Service Structure

Services -> OpcDA_Connector -> <SG_MOD_DA.....>,Service Group (where <SG_MOD_DA.....> is the name of the MOD OPC Service Group)

Select the Service Group Definition Aspect. On the Special Configuration tab Allow Parallel Redundancy should be unchecked.

Do not Enable the service providers at this time. Run the MOD Importer first and then enable the service providers. This will keep objects out of the Lost and Found structure.
6. Run Section 3, MOD Importer and, if necessary, the Phase Importer before continuing.

Tag objects are imported using the MOD Importer as described in the next section. A Connectivity Server for MOD 300 can handle multiple tags and logs. Refer to the *System 800xA, System Guide Technical Data and Configuration (3BSE041434*) instruction for more information. Multiple Connectivity Servers can be created. Each can be assigned to handle a portion of the MOD database as described in Assigning Multiple Connectivity Servers on page 66.
For the MOD 300 Batch Server, it is necessary to run the Phase Importer operation.

Enable the MOD OPC Service Groups

The MOD OPC DA and MOD OPC AE service groups were created within the Service Structure and now have to be enabled, Figure 5. By default, these service providers were not enabled and must be enabled after running the MOD Importer. If not enabled, MOD templates and faceplates (regular and reduced) will not work.

1. Within the Service Structure, open each of the OPCDA Service providers as follows:
   a. Open the OpcDA_Connector, Service
   b. Open a SG_MOD_DA_objectname, Service Group

Figure 5. Enable MOD OPC Service Groups
2. Select a service group and then select the Service Group Definition aspect.
3. Select the Configuration tab, click **Enabled** and **Apply** to enable the service.
4. From the Plant Explorer Workplace, select the Service Structure and open each of the OPCAE Service providers as follows:
   a. Open the EventCollector, Service
   b. Open a SG_MOD_AE_objectname, Service Group
5. Select a service group and then select the Service Group Definition aspect.
6. Select the Configuration tab, select **Enabled** and **Apply** to enable the service.

**Enable the MOD 300 Batch OPC Service Groups**

Optionally, enable the MOD 300 Batch OPC DA service for systems supporting 800xA Batch.

1. Within the Service Structure, open each of the OPCDA Service providers as follows:
   a. Open the OpcDA_Connector, Service
   b. Open a SG_MOD_DA_MODBatch_DA_objectname, Service Group
2. Select a service group and then select the Service Group Definition aspect.
3. Select the Configuration tab, click **Enabled** and **Apply** to enable the service.

No changes are required to the AdvMODBatchOPCServer registry settings to further define the data access configuration. All values should be left at their defaults.

**Complete the Service Structure Set Up**

1. From the MOD AE MOD OPC, service group, Service Group Definition aspect, select the Special Configuration tab, **Figure 6**, and the select the **ABB MOD OPC Alarm Event Server**. The default collection mapping is MOD300Alarms. Click **Apply**.
Section 2  Administration  Complete the Service Structure Set Up

2. Open the Service Structure \ Services \ Time, Service object and select the Service Provider Definition aspect for a node that is not a Connectivity Server. This function is enabled by default and does not need to be changed on the Connectivity Servers. However, on all client nodes, uncheck the Enabled box in the Configuration tab and click **Apply**.
Section 3  MOD Importer

Introduction

The MOD Importer populates the Control Structure of the Aspect Directory with object configuration data built using AdvaBuild. To use the MOD Importer, the following is required:

- All 800xA System software and 800xA for MOD 300 software must be loaded.
- An AdvaBuild for Windows 3.3/x or later project. Editing of MOD 300 Control configuration is done in AdvaBuild.
- Access to a working Aspect System is required.

Full or incremental additions to the Aspect Directory are supported. If an object already exists in the Aspect Directory it will not be overwritten.

Application Description

This section describes the MOD Importer application.

About MOD Importer

The MOD Importer application is packaged with the 800xA for MOD 300 software. The application may be invoked from any 800xA for MOD 300 node. This tool is used to handle the importing of objects, created with AdvaBuild on a MOD Engineering Station (ES), into the 800xA System aspect directory.
Importing MOD Object Types

The tag importer application imports only applicable MOD tag object types. The importing activities include:

- Import a single tag object starting from any object type (parents objects are required for proper support). For example, the node object that is a message source needs to be imported to get messages.
- Import multiple tag objects starting from any object type.

MOD OPC Group Assignments

The following MOD_* objects are assigned to a MOD OPC DA Service Group. The service group is only selectable at the MOD_DB level. Only one service group is supported at this time:

- MOD_AC410, MOD_AC460, MOD_AC460MOD, MOD_ADVANT_D2D, MOD_AREA, MOD_BUC, MOD_BUM, MOD_CCF,
- MOD_CCF_CONTIN_LOOP, MOD_CCF_CONTROL_LOOP,
- MOD_CCFDEVICE_LOOP, MOD_CCF_PID_LOOP, MOD_CNTRLLER,
- MOD_CONSOLE, MOD_CONT_SS, MOD_CTRL_BLOCK, MOD_D2F,
- MOD_DB, MOD_DCN_DCN, MOD_GENERICD, MOD_LL_CNTR,
- MOD_LL_CNTR_GRP, MOD_LL_DEV, MOD_LL_I_O,
- MOD_LL_IO_GRP, MOD_LL_REG, MOD_LL_REG_GRP,
- MOD_LL_TIMER, MOD_LL_TIMER_GRP, MOD_UNIT.

TLL ad hoc object support allows a TLL device to be imported and not the TLL elements that are contained by that device (no object is created). Timers, registers, counters and I/O are called as needed.

General Constraints

Running multiple instances of the MOD Importer is not recommended. This is not enforced by the software.

The MOD 300 Connectivity Server can support up to 6,000 configured objects (tags and logs).

Create a backup of the Aspect Directory database to be able to restore the initial content.
Stopping an import causes the import to roll back to the previous commit point (will not undo changes already made by the importer).

In addition to the illegal characters in AdvaBuild, the 800xA system does not accept the + or / character. These are illegal characters for a tag name on a MOD object.

**Loop Type Change**

To change the loop type in an 800xA for MOD 300 System using AdvaBuild (by adding an A/M or PID FCM to loop that had neither), the correct data will not be displayed in 800xA until nb.store is deleted, the Loop Type in 800xA is changed, and each Connectivity Server node restarted.

The nb.store file can be found by using the Search function in Windows Explorer.

Include hidden directories when doing the search.

If this procedure is performed, it is assumed that you have deleted and imported the loop/tag. Therefore, you will need to redeploy graphics referencing this loop and you must **Check Unresolved Differences**. You will also need to update the Environment Structure/Displays referencing this loop by deleting the loop and then adding the loop.

**Using MOD Importer**

This section describes how to use the MOD Importer application.

**Application Interface**

The MOD Importer application is used to populate the Aspect Directory with tag information which exists within an AdvaBuild server node. Only objects relevant to a MOD 300 node are shown. The MOD Importer application handles each MOD object type by placing them under the OPC DA Service Group.

MOD Importer requires the user to be logged on as part of a group that has local administrator privileges for proper access rights.
User Interface

The MOD Importer Navigator display provides a window into the application. Use the menu commands and toolbar to perform the tasks associated with this application.

Application Start-up and Exit

The MOD Importer application can be run from any node with 800xA for MOD 300 installed.

To start the MOD Importer do the following:

1. In the Control Structure, select the MOD OPC Server Network object and then select the importer aspect.

   A login window appears, Figure 7.

   Figure 7. MOD Import Login

2. Enter or select, if previously entered, a valid server name for an AdvaBuild node and then tab to the Project and User Names for that server.

3. Enter the valid Password for the configuration User Name.

   If the login is invalid, a message box stating that Cannot connect to server with User Name and Password entered! is displayed. If the host is not properly attached, a message box stating that Error connecting to the server
is displayed. If valid, the navigator window appears, Figure 8.

The 800xA logged on user must be listed as a member of the AdvaBuild Users Group in order to connect to AdvaBuild and import the tag data.

4. Select **File > Exit** or close the Navigator window. The application closes.

**Open Another Project**

Select **File > Open** on the navigator window to open the login window. This provides access another server or project. Verify that the currently open database should be closed when prompted.

**Navigator Display**

The Navigator display is a tree view containing the AdvaBuild configuration data starting at the project template (this is a read-only version). The display is opened after login.

The status line, along the bottom of the display, contains the server name, project name, user name, and a status indicator. The status line is enabled/disabled using the **Status Bar** command on the **View** menu.

The Toolbar provides easy command access to the Open, Import and Find commands and is enabled/disabled using the **ToolBar** command on the **View** menu.
Figure 8. MOD Importer Navigator Display

The tree view is color coded to indicate the status of an object. Black text indicates ignore object status, red text indicates object checked and imported as an entry in the Aspect Directory, and blue text indicates object checked and not imported. The object status is set in the view options as described in Setting MOD Importer Options on page 41.
Import MOD Objects and Data

1. Select any object from Navigator window, Figure 8.

2. Select File > Import. Import dialog, Figure 9, appears with selected object in “Import from” field and the installed server name in the “to System” field.

3. Single object selection type. This selection is used to import one object. The parent object must have been imported.

4. When Single Object is not selected, the object and all children objects are imported.

5. Select Import button. An Import progress screen is displayed to show the type and status of the import, Figure 10. A log file is created to document the results of the import.
Figure 10. Importer Progress

If the system has been previously run or is running and objects were placed in the Lost and Found structure of the 800xA system by the Alarm and Event service, then they will be removed, Figure 11. Restart alarm and event services as directed.

Figure 11. Lost and Found Objects
Find MOD Objects

On the MOD Importer display, select **View > Find** (Ctrl+F) to get the Find display, Figure 12.

![MOD Importer Find Display](image)

**Figure 12. MOD Importer Find Display**

**Using Find**

Use the fields under each tab to limit the search as described below.
Name and Location
Enter in the Named: field the name of a specific Object Id or use wildcard characters to find a similar group of names. Use % to match any group of characters and use _ to match and single character.

Select the Look in: field to limit the search to a particular part of the navigator tree. Use the Browse button to navigate through the tree to the object where the search is to begin.

Date
Select All Objects to ignore the date.
Select Objects Created or Objects Modified to set either a date range (Between) or prior to the current date (during the previous month(s) or day(s)).

Advanced
Select one of the following to find objects that meet the stated condition:

- Ignore Operate IT Object Status.
- Objects don’t exist in Operate IT.
- Objects exist in Operate IT.

Find Now
Select Find Now to start search based upon the configuration of each tab in the Find dialog. The results are listed on the Find form. Use the context menu on an object to Import (see below) the object or Show Ancestors (see Show Ancestors on page 39) of the object.

New Search
Select New Search to reset the search parameters except for the Advanced tab.
Import

Select from the list of found objects and then select **Import** to import the objects as described in *Import MOD Objects and Data* on page 35. Use the Ctrl or Shift key along with the left mouse button to select individual objects or a sequence of objects. The import will fail if the parent is not already imported.

Browse Form

Use the **Browse** form, *Figure 13*, to navigate through the object tree and select an object used to determine where a search is to begin.

![Browse Form](image)

*Figure 13. Browse Form*

Show Ancestors

The show ancestors form, *Figure 14*, provides a way to view where a found object exists in the database hierarchy. The display is accessed from the results list on the find display by right clicking the object and selecting **Show Ancestors**.
View MOD Importer Log File

On the MOD Importer display, select View > Log File to open a list of available log files. The open dialog defaults to the last known file location. After selecting a log file, the log file is displayed in a Notepad window, Figure 15.

Using Log File

Use the standard Notepad functions to find information (Edit > Find), to print (File > Print) or exit the Log File.
Log File Results

Typically the log file contains the following information.

- **Objects**: Log file contains start and end time and date, and the total number of objects imported.
- **Object with no parent imported**: Error dialog stating that the object cannot be imported, since its’ parent does not exist.

Setting MOD Importer Options

Select View > Options on the MOD Importer display to:

- Set the tree view tool tip.
- Set the checking of object status.
• Ignore or process TLL elements.

**Tree View Tool Tip**

The tool tip option displays the object type and comment associated with an object as the cursor is placed over the object. This can be turned off by selecting None or enabled in three variations as shown in Figure 16.

**Object Status**

The tree view is color coded to indicate the status of an object as follows:

- **Black Text**  
  *Ignore object status* selected. Using this option allows the tree to be viewed much quicker.

- **Red Text**  
  *Check object status* selected. Object checked and imported as a MOD 300 object into the Control Structure.

- **Blue Text**  
  *Check object status* selected. Object checked and not imported.

The Check System pull down is used to select a System to run a comparison against (currently only one system is supported on an Aspect Server).

**TLL Elements**

This option supports TLL Ad Hoc objects which reduces the number of objects when **Ignore TLL Elements** is selected (TLL elements will not be imported). To import TLL elements, select **Process TLL Elements**.
Figure 16. MOD Importer Options Display
Section 4 MOD Environment Importer

Introduction

The MOD Environment Importer populates the Aspect Server with MOD Area, MOD Group, and MOD Block objects from a structured text file. MOD 300 CDPs can take advantage of the ENVDUMP utility which generates a text file of the Environment. The Environment Builder (Advant application on HP-UX machine) can also be used to generate a print file (File > Print to File). An ASCII file is generated into /home/operator <environment name>. Print without feedback from the Environment Builder interface. These structured files are best viewed in Wordpad.

Application Description

The MOD Environment Importer application is included with the 800xA for MOD 300 software. It imports environment, area, group and block objects from a structured text file generated by the Environment Builder or MOD 300 CDP.

Using MOD Environment Importer

This section describes how to use the MOD Environment Importer application.

Application Requirements

800xA for MOD 300 software must be running to use the MOD Environment Importer. The importer is run from any MOD 300 client.

To use this application, log on as part of a group that has configuration access rights.
Prerequisites

Before using the MOD Environment Importer, make sure that the tag objects referenced by the areas to be imported exist in the database. Use the Tag Status display (see MOD Object Tag Status on page 51) to identify objects that do not exist in the database and what will not be imported from the environment.

The Environment Structure is matched by IDs to the imported Control Structure. If the Control Structure is deleted, then delete the Environment Structure also. Otherwise, the Environment Structure will not match with the Control Structure when it is imported again (object must match with the control structure to work).

Make sure the names GROUP1, GROUP2, GROUP3 are not used for any group names.

Duplicate Areas will not import. It is possible for the ENVDUMP utility to create duplicates in the structured text file if a screen dump is output twice for example.

Application Start-up and Exit

Run the MOD Environment Importer application from any 800xA for MOD 300.

Startup

To start the MOD Environment Importer do the following:

1. Open the path: C:\Program Files(x86)\ABB 800xA\MOD 300\Connect\bin.
2. Open the MODEnvirImporter.exe application.
3. Use the Open window to open a structured text file.

   The structured text file contains the information necessary to display the tree view of the environment in the Navigator window. The ENVDUMP utility on a MOD 300 Console or the Print to File function in the Environment Builder is used to generate the text file.

4. This opens the MOD Environment Importer display with the Navigator window for the selected file, Figure 17.

Exit

Do either one of the following to exit from the current login.
• Select **File > Open** on the navigator window to open another file. Verify that the currently open database should be closed when prompted.

• Select **File > Exit** or close the Navigator window. The application closes.

**User Interface**

The MOD Environment Importer, Figure 17, is the main window into the application. Use the menu commands and toolbar to perform the tasks associated with this application. The Navigator window provides a tree view of the loaded environment.

The status line, along the bottom of the display, contains the file name, environment name, and a status indicator. The status line is enabled/disabled using the **Status Bar** command on the **View** menu.

The Toolbar provides easy command access to the Open, Import and Tag Status commands and is enabled/disabled using the **ToolBar** command on the **View** menu.
The Navigator window is a tree view containing the loaded environment configuration data as a list of areas, groups and tags. The tree view is color coded to indicate the status of an object. Red text indicates object checked and imported as an entry in the Aspect Server and blue text indicates object checked and not imported. Black text is used when the object status is ignored. The object status is set in the view options as described in Setting Options on page 50.
A tree view tool tip is shown when the cursor is placed over the object. The tool tip shows the location of the tag within the area or group of status blocks or rows. The help notation for a tag in an Indicator block in row 1 position 1 is I:1:1, Figure 18. Likewise, a tag associated with a control block in position 2 is C:2. Tool tips are also shown for Area, Group 1, Group 2 and Group 3 locations.

Figure 18. Navigator Tag Tool Tip
Setting Options

Select View > Options to set the checking of object status, Figure 19. Changing the status will cause the Navigator display to refresh.

![Options dialog]

**Figure 19. MOD Importer Options Display**

The tree view is color coded to indicate the status of an object as follows:

- **Black Text**  
  *Ignore object status.* Using this option allows the tree to be viewed much quicker. No checks are done.

- **Red Text**  
  *Check if object exists.* Object checked and imported as an entry in the MOD 300 Environment Structure.

- **Blue Text**  
  *Check if object exists.* Object checked and not imported into the MOD 300 Environment Structure.
Import MOD Environment Objects and Data

Before importing, make sure the prerequisites for imported tags and existing objects are met (see Prerequisites on page 46).

1. Select the Environment or Areas to be imported from the Navigator window. To select multiple Areas, use the Shift + Left Mouse click or the Ctrl + Left Mouse click. Only the Environment and Areas can be selected for import.

2. Select File > Import or the import icon  🌐. The Import window, Figure 20, appears.

   ![Figure 20. MOD Environment Import Display](image)

   The first Area or Environment selected appears in the informational only Import from field.

3. Select the name of the To System from the drop down list.

4. Select the Import button. An Import progress screen displays the type and status of the import. A log file is created to document the results of the import.

   The importer creates Area, Group and Block objects for the selected environment or area(s). However, if the area already exists, it skips the groups during import.

MOD Object Tag Status

On the MOD Importer display, select View > Tag Status to get the Tag Status display, Figure 21. Use the fields under each tab to perform a search as described below. Select File > Exit or the Exit button to leave the Tag Status window.
Figure 21. MOD Environment Importer Tag Status Display

Select one of the following to find objects that meet the stated condition. Using the objects exist or don’t exist option creates a heavy load on the server.

- **Show All** - Lists all tags in the environment.
- **Objects exist in database** - Lists all tags that exist on the Aspect Server that are in the environment.
- **Objects don’t exist in database** - Lists all tags that DO NOT exist on the Aspect Server that are in the environment.

Select **Search** to start search based upon the above selection. The results are listed on the form. Select **File > Save** to save the search results to a text file.
View Log File

Select **View > Log File** to open a list of available log files. The open dialog defaults to the last known file location. After selecting a log file, the log file is displayed in a Notepad window as shown in Figure 22.

![MOD Importer Log File Display](image)

*Figure 22. MOD Importer Log File Display*

Typically the log file contains the following information.

- Log file contains start and end time and date, the name of the starting area object used and the total number of tags assigned to the Area and Group objects created.
- Errors.
- Warnings. This is typically a problem with a tag not existing that is required by that area.
Section 5  MOD 300 TCL Phase Importer

Introduction

The MOD 300 TCL Phase Importer uses sequence names from the MOD Engineering Station TCL Server library to populate the Control Structure in the 800xA system with MOD Phase objects. MOD_UNIT objects must already be imported.

Application Description

The MOD 300 TCL Phase Importer application is included with the 800xA for MOD 300 software.

Using MOD TCL Phase Importer

This section describes how to use the MOD Phase Importer application.

Application Requirements

To use the importer, the following is required:

- All 800xA System software and 800xA for MOD 300 software must be loaded.
- An AdvaBuild for Windows 3.3/x project must be available. Editing of MOD 300 TCL is done in the TCL Builder of AdvaBuild.
- Access to a working Aspect System is required.
- Log on to the computer as part of a group that has configuration access rights to the 800xA system.

Full or incremental additions to the Aspect Directory are supported. If an object already exists in the Aspect Directory it will not be overwritten.
Prerequisites

Before using the MOD Phase Importer, make sure that the sequences to be imported exist in the TCL Builder Server library and that MOD_UNIT objects are already imported. MOD_UNIT objects include the Batch Equipment aspect by default.

Application Start-up and Exit

Run the MOD Phase Importer application from any 800xA for MOD 300 client.

To start the MOD Phase Importer do the following:

1. Go to the path:
   C:\Program Files(x86)\ABB 800xA\MOD 300\Batch\bin.

2. Run the MOD300_Phase_Importer.exe application.
   A login window appears, Figure 23.

   ![Figure 23. MOD Import Login](image)

3. Enter or select, if previously entered, a valid server name for an AdvaBuild node and then tab to the Project and User Names for that server.

4. Enter the valid Password for the configuration User Name.
   If the login is invalid, a message box stating that **Cannot connect to server with User Name and Password entered!** is displayed. If the host is not properly attached, a message box stating that **Error connecting to the server** is displayed. If valid, the navigator window appears, Figure 24.
Do either one of the following to exit from the current login.

- Select **File > Open** on the navigator window to access another server or project. Verify closing the current file.
- Select **File > Exit** or close the Navigator window. The application closes.

**User Interface**

The MOD300 TCL Phase Importer, Figure 24, is the main window into the application. Use the menu commands and toolbar to perform the tasks associated with this application. The Navigator window provides a tree view of the unit and sequence names. The tree view has a color code option to indicate the status of an object. The view options are described in Setting Options on page 58.

![Figure 24. MOD Phase Importer Navigator Display](image)

The status line, along the bottom of the display, contains the server name, project name, user name, and a status indicator. The status line is enabled/disabled using the **Status Bar** command on the **View** menu.

The Toolbar provides easy access to the Open and Import commands and is enabled/disabled using the **ToolBar** command on the **View** menu.
Setting Options

Select **View > Options** to set the checking of object status, Figure 25. Changing the status will cause the Navigator display to refresh.

![Options](image)

*Figure 25. MOD Phase Importer Options Display*

When the **Ignore Operate IT object status** is selected, the tree view will show in black only. Using this option allows the tree to be viewed much quicker. No checks are done.

When the **Check if Operate IT object exists** is selected, the tree view will be color coded red or blue (see below) to indicate the status of an object. Checking the object status takes time especially when there are many phases.

<table>
<thead>
<tr>
<th><strong>Red Text</strong></th>
<th>Object checked and imported as an entry in the MOD 300 Control Structure.</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Blue Text</strong></td>
<td>Object checked and not imported into the MOD 300 Control Structure.</td>
</tr>
</tbody>
</table>
Import MOD Phase Objects

Before importing, make sure the prerequisites are met.

1. Select the phases to be imported from the Navigator window. To select multiple phases, use the Shift + Left Mouse click or the Ctrl + Left Mouse click. Selecting a unit selects all the phases in that unit.

2. Select File > Import or the import icon 📌. The Import window, Figure 26, appears.

![Figure 26. MOD Phase Import Display](image)

The first phase selected appears in the informational only Import from field.

3. Select the Import button. An Import progress screen is displayed to show the type and status of the import. A log file is created to document the results of the import.

The importer creates phase objects for the selected phases. However, if the phase already exists, it skips the phase during import.

When a MOD_PHASE is imported, the TCL sequence name is assigned to the MOD_Phase object and to the Value field of the SourceName in the Generic Properties aspect. The object name can be changed (it does not have to be identical to the sequence name). However, the SourceName must match the TCL sequence name. Each MOD_Phase within a MOD_Unit must have a name that is unique within that MOD_Unit.

If a phase is executed on a unit in parallel, the alias identifier must be also set. See Appendix C, Batch Support.
View Log File

Select **View > Log File** to open a list of available log files. The open dialog defaults to the last known file location. After selecting a log file, the log file is displayed in a Notepad window.

Typically the log file contains the following information.

- Log file contains start and end time and date, the name of the starting area object used and the total number of phases assigned to the objects created.
- Errors.
- Warnings. This is typically a problem with a phase not existing that is required.
Section 6  Configuration

Introduction

MOD 300 tags can be added using the procedures described in this section. Refer to *800xA System Configuration (3BDS011222*) for an overview and specific information on system related configuration topics such as security, system definition, project and system structure, and alarm, event and report building. Information on graphic displays can be found in *800xA Engineering Process Graphics (3BSE049230)*.

After starting the 800xA system, the remaining activities are optional and are provided to allow customizing of the Control Structure, permissions, OPC Server, graphics, alarm colors and NLS text.

Third party applications using an OPC Client Interface must use OPC Server for MOD 300. Refer to *800xA for Mod 300 configuration (3BUR002262*) for configuring a third party OPC Client Interface to MOD 300.

Start the 800xA System

As an administrator, use the Configuration Wizard to start each node of the 800xA System.

Before starting the system, ensure that **Autostart System on Windows startup** is checked in the System software User Setting dialog of the Configuration Wizard. Ensure this for each node. The MOD OPC Server requires this setting to start properly.

1. From the ABB Start Menu, select **Configuration Wizard**, select **System Administration** and then click **Next**.
2. Select the System Name to be started and click **Next**.
3. Select **Systems** to perform the Action on the system and click **Next**.

4. Select **Start** as the action to be performed and click **Next**.

5. Click **Finish** to apply the settings.

After the system is started, open the Plant Explorer Workplace by selecting it from the ABB Start Menu to perform the activities described in the rest of this section.

Do not have two plant Explorers open at the same time when making changes.
This is not supported.

---

**MOD OPC Server Configuration**

Prior to the release of 800xA for MOD 300 SV4.1 RU3 and SV5.0 SP1, the MOD 300 OPC DA Server retrieved data at varying subscription rates. Thus subscription rates varied between graphic displays, faceplates, and history data.

With the original implementation of the 800xA for MOD 300 OPC DA server with support for varied subscription rates, users are not required to perform any specific configuration for trend data quality presentation or history data because the composite attribute MEASURE_COMP was automatically supplied even if the log or trend was configured to get the loop’s MEASURE attribute. **Subscription rates >= to 4 seconds** were automatically set to retrieve the *_COMP* composite attribute (* is equal to either MEASURE, SETPOINT or OUTPUT). It was assumed that these longer subscription rates would be used for historical recording. Using *_COMP* provided the data quality attribute and alarm attribute that enabled bad data quality (DQ) and alarm presentation on the trend displays.

A Fixed Rate MOD 300 OPC DA Server was developed to improve the performance of 800xA for MOD 300. It is now the defaulted installation with a two second subscription rate. The user can change the subscription rate in the Fixed Rate MOD 300 OPC DA Server. In addition, the user needs to configure use of the composite attribute for trend data quality presentation or history data. Reference sections below.
Fixed Rate MOD 300 OPC Server Configuration

The Fixed Rate MOD 300 OPC DA Server was developed to improve the performance of 800xA for MOD 300. It is the defaulted installation.

The default setting is Fixed Rate at 2 seconds (2000 ms).

To modify the configuration of the Fixed Rate MOD 300 OPC Server modify the registry settings as identified below.

1. Locate the registry settings at:
   \HKEY_LOCAL_MACHINE\SOFTWARE\Wow6432Node\ABB\AdvOPCDAServer\config

2. Modify (re-configure) the two registry keys related to the Fixed Rate MOD 300 OPC Server.

<table>
<thead>
<tr>
<th>Registry Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Fixed Rate Server</td>
<td>• 0 = Fixed Rate server is not used.</td>
</tr>
<tr>
<td></td>
<td>• 1 = enabled. Fixed Rate server is used. Default value is 1.</td>
</tr>
<tr>
<td>Fixed Rate for OPC DA Server (ms)</td>
<td>&gt;=2000 ms. Default value is 2000 ms</td>
</tr>
</tbody>
</table>

3. Restart Process Administration System (PAS) software for changes to be implemented.

Refer Customized Data for Backup for more detailed information.

Composite (_COMP) Configuration for MOD Trend Data Quality and History Data

Follow one of the options below to configure using the composite attribute (*_COMP) for history logs or trend display data quality presentation.

1. Directly specify the composite attribute (*_COMP).

   Composite attribute applies to MEASURE_COMP, SETPOINT_COMP and OUTPUT_COMP.
2. Automatically use the composite attribute (*_COMP) by changing registry settings.

Properly specify the registry attributes for “Use Composite Update Time Change” and “Composite Update Time Change” (ms).

These registry attributes assume that the Fixed Rate OPC server attribute is enabled and the “Fixed Rate for OPC DA Server” attribute is >= 2000 ms.

Access these attributes in the following registry location:

\[HKEY_LOCAL_MACHINE > SOFTWARE > Wow6432Node > ABB > AdvOPCDAServer > config\]

<table>
<thead>
<tr>
<th>Registry Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Fixed Rate Server</td>
<td>• 0 = Fixed Rate server is not used.</td>
</tr>
<tr>
<td></td>
<td>• 1 = enabled. Default value is 1.</td>
</tr>
<tr>
<td>Fixed Rate for OPC DA Server (ms)</td>
<td>&gt;=2000 ms. Default value is 2000 ms</td>
</tr>
<tr>
<td>Use Composite Update Time Change</td>
<td>• 0 = Disabled; composite attributes not used unless specified directly using *_COMP</td>
</tr>
<tr>
<td></td>
<td>• 1 = Enabled; uses *_COMP when subscription rate is greater or equal to Composite Update Time Change (ms) value.</td>
</tr>
<tr>
<td>Composite Update Time Change (ms)</td>
<td>Threshold at which *_COMP is used; works in conjunction with the Use Composite Update Time Change key; this should be set to a value greater than the “Fixed Rate for OPC DA Server”. For example, Fix Rate for OPC DA Server set to 2000ms, value for Composite Update Time Change should be 2001ms or greater.</td>
</tr>
</tbody>
</table>

**Note**: History Logs should be configured such that the sample interval is >= Composite Update Time Change.

**Example: Use *_COMP attributes when collecting history data**

Set the registry settings and history log configuration as follows:

- **Use Composite Data Update Time Change** set to enabled (1)
• **Use Fixed Rate Server** set to enabled (1)
• **Composite Update Time Change** (ms) greater than the Fixed rate for OPC DA server (ms)
• History log sample interval is configured greater than or equal to the Composite Update Time Change (ms) rate

With this configuration, the history data collected will automatically include *COMP attributes. There is no need to explicitly define *COMP for history data collection.

**Define the MOD OPC Data Source**

Open the Control Structure, select the MOD_DB object and then select the OPC Data Source Definition aspect as shown in Figure 27. From the Connectivity tab, select a service group using the pull-down select list as shown. Click **Apply** when done.

*Figure 27. OPC Data Source Definition*
Assigning Multiple Connectivity Servers

The MOD Batch OPC Data Source must be configured when using 800xA for MOD 300 with 800xA Batch. Configure the MOD Batch OPC Server Definition aspect under the MOD_DB object in the Control Structure to reference the MOD 300 Batch OPC DA Service Group defined in the Service Structure by selecting the Service Group from the Connectivity Tab in the MOD Batch OPC Data Source Definition aspect.

When creating a new Data Source Definition Aspect check the Data Source Key and if needed, modify the Data Source Key to match the default ones delivered on the MOD_DB object. There is a unique key for the Batch OPC DA server and a unique key for the OPC DA Server.

For systems that also include Batch Management, when creating Data Source Definition Aspects always create them in a pair that includes both the Batch OPC DA and MOD 300 OPC DA definitions.

Assigning Multiple Connectivity Servers

A Connectivity Server for MOD 300 can handle a limited number of tags and logs. Force a Connectivity Server to handle a portion of the MOD database by copying the MOD OPC Data Source aspect into a specific area (objects search up for the proper connectivity server). To do this, copy the MOD OPC Data Source aspect from the MOD_DB onto an appropriate MOD Area Object (for example). After copying the MOD OPC Data Source aspect, set the Service Group to be a different Connectivity Server (pair) and click Apply. MOD objects that are children of that area will then use the assigned Connectivity Server.

Alarm Routing with Multiple Connectivity Servers

Alarm routing must be set up to route a specific area, as defined in AdvaBuild, to the same area as defined in the Connectivity Server that has the OPC Data Source definition for that area only. Routing alarms to multiple Connectivity Servers will cause alarms and diagnostics to not be synchronized. While all alarms for one area need to be routed to one Connectivity Server, data access (OPCDA) can be split between two Connectivity Servers.
When rebooting a Connectivity Server node pair where MOD 300 messages/events are routed, wait until the primary Connectivity Server node is fully booted and active (view AdvOPCAEServerTrace.log for a line `synchronize list: done`) before rebooting the standby Connectivity Server node to prevent an alarm mismatch.

It is not possible to define routing for Diagnostic Alarms. Diagnostic Alarms are routed to all connectivity server pairs even though the system may be configured to route alarms/messages (For example. CCF, TCL) to a single connectivity server pair as recommended. This will result in duplicate diagnostic alarms being presented in those systems with multiple connectivity server pairs. These diagnostic alarms, including the duplicates, can be cleared by acknowledging the alarms.

**Running MOD 300 Utility Aspect Report**

The MOD 300 Utility aspect as shown in Figure 28, generates a summary report that displays how many MOD tags are assigned to a MOD OPC Data Source and provides a total number of tags and logs assigned to a Connectivity Server. The MOD 300 Utility is an aspect of the MOD_DB object found in the Control Structure. To create a report, select the aspect and then select the **Run Report** button. The results of the report are displayed in the report window and can be copied using the context menu.

MOD 300 Utility report includes a summary of the number of tags by type and total number of tags for each MOD OPC Data Source as well as a total for the Connectivity Server. The report shows totals for the following tags and logs:

- MOD_CCF_CNTRL_LOOP.
- MOD_CCF_CONTIN_LOOP.
- MOD_CCF_PID_LOOP.
- MOD_CCF_DEV_LOOP.
- MOD_UNIT.
- MOD_LL_DEV.
- Log Configuration.
Running MOD OPC Statistics Aspect

Figure 28. MOD 300 Utility

The MOD OPC Statistics aspect supports application development/system loading calculations by showing the number of Reads, Subscriptions, Updates and Writes as well as other related data (such as Update Rate, Value and Quality).

Open the aspect from:

Service Structure > Services > OpcDA_Connector, Service > SG_MOD_DA_MOD OPC Server, Service Group > OPCDA_Provider_pcname, Service Provider.

and then select the MOD OPC Statistics Aspect.

When Subscribe for live data is checked, the active numbers for information being handled by the MOD OPC Server are displayed. For example, an active PID faceplate may show 31 live subscriptions.
Alarm Event Configuration

Alarm and Event configurations in the registry will be changed upon upgrade. Refer to Appendix F, Customized Data for Backup for more detailed information.

An alarm mismatch between the primary and redundant Connectivity Server nodes of a redundant pair(s) occurs after rebooting the pair. Both nodes try to synchronize with the controllers and alarms may transition while the redundant server is starting, resulting in alarms having differing timestamps. When rebooting a Connectivity Server node pair, wait until the primary Connectivity Server node if fully booted and active (view AdvOPCAEServerTrace.log for a line synchronize list: done) before rebooting the standby Connectivity Server node to prevent the alarm mismatch.

Data Access Configuration

Data Access configurations in the registry will be changed upon upgrade. Refer to Appendix F, Customized Data for Backup for more detailed information.

MOD Alarm and Event Logger Configuration

The following MOD Alarm and Event Loggers are delivered as default aspects in the Library structure. The filters for the logger aspects are the same as for their equivalent list aspects.

- MOD CCF Alarm Logger (MOD CCF Alarm).
- MOD Diag Logger (MOD Diagnostic Operator).
- MOD Event Logger (MOD Parameter Change, MOD User Acknowledge, MOD User Log On/Off).

These default aspects allow a custom logger to be created.

1. In the Service Structure under Alarm Logger, Service \ Basic, Service Group, select the Service Group Definition aspect.
2. On the Special Configuration tab, set the Logger Configuration Name to one of the above MOD loggers.
3. Select the Service Provider Definition aspect for a node with a logger and on the Special Configuration tab, set the Logger Printer Name to a printer defined for that node.

4. Go to the Admin Structure under Administrative Objects \ Domains \ Domain and select the Audit Trail Config aspect.

5. Set the Audit Trail active check box and also set the Log generic audit events and the Log selected audit event classes. Set Audit Event classes as desired.

6. Customize the Alarm and Event List Logger Configuration aspect as required.

The MOD Parameter Change filter does not report changes to the Loop Detail display (ENG DEADBAND, OUTPUT DEADBAND and DEVIATION DEADBAND), the TCL Sequence Detail display (Pause Step), and all I/O Displays.

**Audit Trail and Parameter Change Messages**

Depending on the display, there may be multiple parameter change messages sent to the Audit Trail. The duplicate message will not include the logged on user. The following displays can have duplicate messages recorded in the Audit Trail:

- CCF faceplates.
- TLL faceplates.
- Graphics.
- Runtime Templates.

Parameter change requests from these displays will cause an Audit Trail message to be recorded directly to the Information Manager node and these messages will include the logged on user ID. These parameter change requests are sent to the 800xA for MOD 300 controllers which generate parameter change messages that may also be routed to the Information Manager through the connectivity servers. This Parameter change message will not include the logged on user ID but will show Administrator as making the parameter change because the logged on user ID is not passed to the 800xA for MOD 300 controller from these displays.

The following displays will pass the logged on user ID to the controller and parameter change messages generated by the controller will include the user ID. These displays also send an Audit Trail message including logged on user ID directly to the Information Manager node:
• Loop FCM.
• Loop Detail.
• TLL Displays.
• TCL Displays.
• I/O Displays.
• Diagnostic Displays.

If the MOD 300 parameter change messages are routed through the connectivity server nodes, then there will be duplicate messages in the Information Manager. In order for the Information Manager to receive parameter change messages, the Audit Trail must be configured. For all cases the parameter change message from the controller will only be sent if the controller is configured to route parameter change messages to the Connectivity servers (such as the Logger on the RTAB).

To get all Operator Action Tracking Events from the 800xA system, use the AuditEvent_OperatorAction filter.

Control Structure Configuration

The MOD Importer is the primary tool to create MOD tag objects within the 800xA System Control Structure. The option to create MOD objects manually is also provided as described here. In addition, the ability to define access rights and to build graphic displays with MOD specific elements is provided in the information referenced here.

• Control Structure Configuration Using MOD Templates on page 71.
• Special Permissions for MOD Objects on page 76.
• Creating a MOD CCF Loop Object on page 81.

Control Structure Configuration Using MOD Templates

The AdvaBuild project control structure is represented as a MOD_DB (Advant OCS Database) object in the 800xA System Control Structure. Control structure objects represent the physical and functional characteristics of the control system. For
instance, there are objects that represent Advant Stations, Controller Subsystems, control modules, control loops, and so on.

**Control Structure Relationships**

Objects in the control structure are related by parent-child relationships as shown in Table 1. For instance, a control loop is a child of a Configurable Control Function (CCF) object, which in turn is a child of a control module object. These parent-child relationships are established automatically when objects are inserted in the control structure. Like AdvaBuild, the software has built-in checks so that an invalid relationship between objects cannot be inadvertently established. However, these checks are on relationships only and it is possible to put an object that is physically on another controller into a similar controller (AC1-3 into AC3-3 for example).

Use **New Object** in the Control Structure to create these object types. The tag needs to be a valid tag in the AdvaBuild database for the project.

*Table 1. Control Structure Object Relationships for MOD 300*

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD_DB</td>
<td>MOD_DB is the top of the control structure and represents a specific Advant OCS database.</td>
</tr>
<tr>
<td>MOD_AREA</td>
<td>AREA objects allow the Advant OCS database to be partitioned into one or more areas. Its children are objects that represent subsystems (nodes) in a given area.</td>
</tr>
<tr>
<td>MOD_AC410, MOD_AC460, MOD_ADVANT_D2D, MOD_ADVANT_STATION, MOD_CONSOLE, MOD_CONT_SS, MOD_DCN_DCN, MOD_GENERICD</td>
<td>The AC410, AC460, ADVANT_D2D, ADVANT_STATION, CONSOLE, CONT_SS, DCN_DCN, GENERICD objects are children of an AREA object and represent subsystems (nodes) in a given area. The ADVANT_STATION object is created in the 800xA System in place of the GENERICD object when it is identified as an Advant Station (HP-UX).</td>
</tr>
</tbody>
</table>
Table 1. Control Structure Object Relationships for MOD 300 (Continued)

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD_CCF, MOD_LL_DEV, MOD_UNIT, MOD_BUC, MOD_BUM, MOD_BUM, MOD_D2F, MOD_CNTRLLER, MOD_AC460MOD</td>
<td>The CCF, LL_DEV, UNITMAST, BUC, CNTRLLER, AC460MOD, D2F objects are children of the subsystem-level objects. They establish the physical and functional characteristics of the subsystems.</td>
</tr>
<tr>
<td>MOD_CTRL_BLOCK</td>
<td>The CTRL_BLOCK object is a child of the CCF object and may be used as a parent for LOOP_DEF or DEV_LOOP objects to group related loops. This object is optional in the control structure.</td>
</tr>
<tr>
<td>MOD_CCF_CNTRL_LOOP, MOD_CCF_CONTIN_LOOP, MOD_CCF_PID_LOOP, MOD_CCF_DEVICE_LOOP</td>
<td>The LOOP_DEF object represents a control loop (CNTRL, CONTIN, PID). It is a child of the CCF object. The control algorithm is defined by the Function Class Modules (FCMs) within the LOOP_DEF object such AINPUT, AOUTPUT, MSQR, PID and others. The DEV_LOOP object represents a discrete device. It is also a child of the CCF object.</td>
</tr>
</tbody>
</table>

MOD Templates Supported in the Object Type Structure

The MOD configuration templates provide similar operations and status displays used with AdvaCommand. Within the Object Type Structure is a MOD Control System object type group that contains the templates for the MOD object types listed below. As Control Structure objects are created, objects based on these templates are inserted into the structure. Where dependencies exist, a supporting object is also created.

- MOD_DB_GROUP
  - MOD_DB
  - MOD_AREA
- MOD_ENV_GROUP
- MOD_ENV
- MOD_ENV_AREA
- MOD_ENV_GROUP
- MOD_ENV_BLOCK
- MOD_ENV_BLANK

- MOD_CCF_GROUP
  - MOD_CCF*
  - MOD_CCF_CNTRL_LOOP
  - MOD_CCF_CONTIN_LOOP
  - MOD_CCF_DEVICE_LOOP
  - MOD_CCF_PID_LOOP
  - MOD_CCTRL_BLOCK

- MOD_TCL_GROUP
  - MOD_UNIT
  - MOD_SEQUENCE (refer to MOD_Sequence Object on page 76)

- MOD_TLL_GROUP
  - MOD_LL_DEV
  - MOD_LL_CNTR_GRP
  - MOD_LL_CNTR
  - MOD_LLIO_GRP
  - MOD_LL_I_O
  - MOD_LL_REG_GRP
  - MOD_LL_I_O
  - MOD_LL_REG
  - MOD_LL_REG
  - MOD_LL_TIMER_GRP
  - MOD_LL_TIMER

- MOD_NODES_GROUP
Certain MOD templates are assigned special names, by the MOD Importer only, as listed in Table 2. The default object names are based upon the source MOD template/tag name.

**Table 2. MOD Configuration Templates with Special Object Names**

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Object Created</th>
<th>Object Name</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>CONT_SS</td>
<td>MOD_D2F</td>
<td>name_DF1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MOD_D2F</td>
<td>name_DF2</td>
<td></td>
</tr>
<tr>
<td>BUC</td>
<td>MOD_BUM</td>
<td>name_BUM1</td>
<td>sub device address = 4</td>
</tr>
<tr>
<td></td>
<td>MOD_BUM</td>
<td>name_BUM2</td>
<td>sub device address = 8</td>
</tr>
<tr>
<td></td>
<td>MOD_BUM</td>
<td>name_BUM3</td>
<td>sub device address = 12</td>
</tr>
</tbody>
</table>
MOD_Sequence Object

The MOD_SEQUENCE object supports direct access of a valid MOD sequence. From a MOD_Unit object select New Object. The MOD_SEQUENCE object will show in the dialog. Enter a name of a valid MOD Sequence in the Name field and then click Create. The TCL Sequence Detail display is accessed directly from this object. Other aspects include the TCL Sequence Debug and the TCL SFC displays.

Special Permissions for MOD Objects

MOD objects have a set of predefined permissions which can be set as an allowed or denied action to an object (predefined permission groups use allowed actions to define permissions). The permissions in Table 3 apply to the objects specified. For example, MOD_ALARMCONFIG applies to MOD_CCF_PID_LOOP objects. Refer to 800xA System Security for additional information on permissions and security groups and methods. In general, the order of precedence is to look up and through the structure for a denied permission, continue if no permission is found or until an allowed permission is found.

Edit the primary MOD 300 security group and assign object access rights:

1. Select the MOD_DB object in the Control Structure.
2. Select the Security Definition aspect as shown in Figure 29.

<table>
<thead>
<tr>
<th>Object Type</th>
<th>Object Created</th>
<th>Object Name</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AC460</td>
<td>MOD_D2F</td>
<td>name_DF1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MOD_D2F</td>
<td>name_DF2</td>
<td></td>
</tr>
<tr>
<td>DCN_DCN</td>
<td>MOD_DCN_DCN</td>
<td>name_D2D1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MOD_DCN_DCN</td>
<td>name_D2D2</td>
<td></td>
</tr>
<tr>
<td>ADVANT_D2D</td>
<td>MOD_ADVANT_D2D</td>
<td>name_D2D1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>MOD_ADVANT_D2D</td>
<td>name_D2D2</td>
<td></td>
</tr>
<tr>
<td>CONT_SS</td>
<td>MOD_D2F</td>
<td>name_DF1</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>name_DF2</td>
<td></td>
</tr>
</tbody>
</table>

Table 2. MOD Configuration Templates with Special Object Names (Continued)
3. Select a permission.
4. Set the allowed permissions as required for the group.

Edit other MOD 300 objects using the Security Definition aspect of the object to control access to that object as required.

Figure 29. MOD_DB Security Definition Aspect
### Table 3. 800xA System Permissions for MOD Objects

<table>
<thead>
<tr>
<th>Permissions</th>
<th>MOD Objects / Description</th>
<th>Allowed Security Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>Configure</td>
<td>All objects. Allows configure rights.</td>
<td>System Engineers</td>
</tr>
<tr>
<td>Operate</td>
<td>All objects. MOD objects have their own permissions.</td>
<td>Operators, System Engineers, Application Engineer</td>
</tr>
<tr>
<td>Read</td>
<td>All objects. Allows read rights.</td>
<td>Operators, System Engineers, Application Engineer</td>
</tr>
<tr>
<td>Tune</td>
<td>All objects. Allows tuning rights.</td>
<td>System Engineers</td>
</tr>
</tbody>
</table>
| MOD_SIM_MODIFY      | ADVANT_D2D, CONSOLE, DCN, GENERICD  
Allows SIM annunciator to be turned on.                                                   | Operators, System Engineers, Application Engineer |
| MOD_HARDRESET       | AC460, AC460MOD, BUC, BUM, CNTRLLER, CONT_SS  
Allows HARDRESET and Start Controller functions.                                          | System Engineers                                  |
| MOD_REDUNDANCY      | Allows, BUM, CNTRLLER, CONT_SS  
Allows Failover, Switchback, Establish Redundancy and Terminate Redundancy functions.   | System Engineers                                  |
| MOD_UPGRADE         | AC460, AC460MOD  
Allows RESET for UPGRADE and Switchover functions.                                         | System Engineers                                  |
| MOD_RESETCOUNTERS   | AC410, AC460, ADVANT_D2D, BUC, CNTRLLER, CONSOLE, D2F, DCN_DCN, GENERICD                  | Operators, System Engineers, Application Engineer |
### Table 3. 800xA System Permissions for MOD Objects (Continued)

<table>
<thead>
<tr>
<th>Permissions</th>
<th>MOD Objects / Description</th>
<th>Allowed Security Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD_LOADSEQUENCE</td>
<td>UNIT Load sequences from the Unit Detail display.</td>
<td>System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_LOADRECIPE</td>
<td>UNIT Load recipes from the Unit Detail display.</td>
<td>Operators, System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_SEQCONTROL</td>
<td>UNIT Control sequences from the TCL displays that have sequence control functionality.</td>
<td>System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_SEQDEBUG</td>
<td>UNIT Modify parameters on the TCL Sequence Debug display.</td>
<td>System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_RECIPECONTROL</td>
<td>UNIT Modify Recipe parameters on the Recipe Detail display.</td>
<td>Operators, System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_STEPCONTROL</td>
<td>UNIT</td>
<td>Operators, System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_TLLCONTROL</td>
<td>LL_* (CNTR, DEV, IO, REG, TIMER) Modify parameters on these TLL displays.</td>
<td>Operators, System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_LOADSEGMENT</td>
<td>DEV Load Segments on the TLL Segment display.</td>
<td>System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_SEGMENTCONTROL</td>
<td>DEV Modify parameters on the TLL Segment display.</td>
<td>System Engineers, Application Engineer</td>
</tr>
</tbody>
</table>
### Table 3. 800xA System Permissions for MOD Objects (Continued)

<table>
<thead>
<tr>
<th>Permissions</th>
<th>MOD Objects / Description</th>
<th>Allowed Security Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD_ALARMCONFIG</td>
<td>PID, CNTRL, CONTIN, DEVICE Modify alarm parameters on the Loop Detail display. Also, Alarm Post, Alarm Check, Engineering Deadband, Output Deadband, Deviation Deadband.</td>
<td>System Engineers, Application Engineer</td>
</tr>
<tr>
<td>MOD_LOOPTUNE</td>
<td>PID Modify Tuning parameters on the Loop Detail display (Gain, Reset, Preact).</td>
<td>System Engineers</td>
</tr>
<tr>
<td>MOD_LOOPSETUP</td>
<td>PID, CNTRL, CONTIN, DEVICE Modify Loop values on the Loop Detail display (Phase, Scan Rate, Loop Mode, Trend Rate). Also, Simulation Mode, Override Mode, and Lock State on Device Loop.</td>
<td>System Engineers</td>
</tr>
<tr>
<td>MOD_TRIOCONTROL</td>
<td>AC410, AC460, CNTRL Modify parameters on the TRIO runtime displays. OR (since both can not exist at the same time), CNTRL Modify parameters on the Direct I/O runtime displays (applies only to a SC or Model B Controller)</td>
<td>System Engineers</td>
</tr>
<tr>
<td>MOD_S100CONTROL</td>
<td>AC410, AC460 Modify parameters on the S100 I/O runtime displays</td>
<td>System Engineers</td>
</tr>
</tbody>
</table>
Creating a MOD CCF Loop Object

Use Create Object in the Control Structure to create new MOD objects based upon the MOD configuration templates as shown in Table 2. The tag needs to be a valid tag in the AdvaBuild database for the project. While logical structures are enforced, physical structures are not.

1. Open the MOD_DB object tree in the Control Structure to the location where a new CCF loop object is to be placed.
2. Select the CCF object and use the context menu to select **New Object**.
3. From the **New Object** dialog, select a CCF loop object as shown in Figure 30.

### Table 3. 800xA System Permissions for MOD Objects (Continued)

<table>
<thead>
<tr>
<th>Permissions</th>
<th>MOD Objects / Description</th>
<th>Allowed Security Groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOD_S800CONTROL</td>
<td>AC410, AC460</td>
<td>System Engineers</td>
</tr>
<tr>
<td></td>
<td>Modify parameters on the S800 I/O runtime displays.</td>
<td></td>
</tr>
<tr>
<td>MOD_PROFICONTROL</td>
<td>AC460</td>
<td>System Engineers</td>
</tr>
<tr>
<td></td>
<td>Modify parameters on the PROFIBUS runtime displays.</td>
<td></td>
</tr>
</tbody>
</table>
Enter a valid MOD tag in the **Name** field and then click **Create**.

In addition to the illegal characters in AdvaBuild, the 800xA system does not accept the + or / character. These are for a tag name on a MOD object.

---

**Environment Structure Configuration**

The Environment Importer will set the Environment, Area, Group, Block and Tag configurations within the MOD 300 Environment Structure. Graphics will need to
be configured for the Area Graphic and Group Graphic displays. This section describes how to add or modify environment configurations.

**Configuring a MOD Environment**

- Copy/Paste is not supported in the configuration or groups/areas.
- No support for upper/lower case tags.

The MOD Environment object is created by the MOD Environment Importer when importing an environment. To manually create an environment object,

1. Select the MOD 300 Environment Structure.
2. Click on New Object from the context menu within the object structure white space (minimize all MOD_ENV objects if necessary).
3. Select MOD_ENV from the list of objects, name it and click Create.

Multiple environments can be set up to meet the needs of the Area and Group organizations.

**Configuring a MOD Environment Area Object**

This procedure describes how to add or modify a MOD Area object. The MOD Area object supports Alarm, Status, and Graphic displays. An area is a collection of up to three groups. As such, each area is composed of up to 108 loops (control loops, indicator loops, or a combination of both). This provides a broader view of the process from a single display. To create an Area and assign Groups:

1. Select the MOD 300 Environment Structure.
2. Select an Environment (MOD_ENV) object.
3. Select New Object from the context menu for the MOD_ENV object.
4. Create a MOD_ENV_AREA object (name it and click Create).
5. Select the MOD300 Environment Area Configuration aspect as shown in Figure 31.
6. Use the **Insert**, **Remove**, **Move Up**, and **Move Down** buttons to configure up to three groups into the area. A new Group could also be created from the MOD_ENV_AREA object by selecting New Object and adding a MOD_ENV_GROUP object.

7. Click **Apply** to save the configuration. There is no automatic refresh with these objects and the aspect may need to be refreshed to see the new group.

**Configuring a MOD Environment Group Object**

This procedure describes how to add or modify a MOD Group object. The MOD Group object supports Trend, Alarm, Status, and Graphic displays.
A group is a collection of related loops. A group can have up to 36 loops (control loops, indicator loops, or a combination of both). Generally, a group has up to 12 control loops or up to 36 indicator loops. A control loop tag can be assigned to an indicator position in the group layout if only the basic indicator type information is to be displayed. A tag can be in more than one group.

**Assigning Blocks to a Group**

1. Select the **MOD 300 Environment Structure**.

2. Open an Environment (**MOD_ENV**) and then a **MOD_ENV_AREA** object to select a **MOD_ENV_GROUP** object.

3. Open the **MOD300 Environment Group Configuration** aspect as shown in Figure 32.

4. Use the **Insert**, **Remove**, **Move Up**, and **Move Down** buttons to configure up to twelve blocks into the group. A new Block could also be created from the **MOD_ENV_GROUP** object by selecting New Object and adding a **MOD_ENV_BLOCK** object.

   If all 12 blocks are not assigned and a tag is added to a non-existing block, then all unassigned blocks will be added with the default name of Block.

5. Click **Apply** to save the configuration. There is no automatic refresh with these objects and the aspect may need to be refreshed to see the new block.
Assigning Tags to a Block using Group Layout

Make tag assignments from the MOD300 Environment Group Layout Configuration aspect. Visually, the Group Layout is the preferred display for this task. The MOD_ENV_BLANK object is used as a filler (layout will still show I) and is added as a New Object from the MOD_ENV_BLOCK object.

1. Open the MOD300 Environment Group Layout Configuration aspect as shown in Figure 33.

2. Select I for Indicator or C for Control and use the Browser to locate a tag in the Control Structure as shown in Figure 34 or type in a tag name. A message is displayed if the tag is not in the Control Structure.

3. Click Apply to save the configuration.

Figure 32. MOD300 Environment Group Configuration Aspect
Figure 33. MOD Environment Group Layout
Figure 34. MOD Environment Browse for Tag Name

Assigning Tags to a Block using Block Configuration

4. The MOD300 Environment Block Configuration aspect as shown in Figure 35 can also be used to assign tags. Using the **Insert**, **Remove**, **Move Up**, and **Move Down** buttons to configure one control loop or two or three tag indicator loops into the block.
5. Click **Apply** to save the configuration.

**Graphics**

Graphics are associated with MOD 300 objects by assigning a graphic aspect, or when editing the MOD 300 Area Graphic Display or the MOD 300 Group Graphic Display. Custom graphic displays are built using the Graphic Builder Tool, reference *System 800xA Engineering Process Graphics (3BSE049230)*.

With the System 800xA 5.1 release, Process Graphics 2 is the default. Use of the MOD 300 Group Graphic Display will default to using a Process Graphic 2 Graphic Display Aspect named Group Graphic Element.

To use a Visual Basic aspect with MOD 300 Group Graphic Display in a new installation of an 800xA 5.1 System, the Visual Basic Graphic Display aspect named Group Graphic Element must be overridden on the object. This causes the MOD 300 Group Graphic Display to use the VB Group Graphic Element. Note that
the Visual Basic graphics extension must also be installed and loaded in the system (800xA for MOD 300 VB Graphics Extension).

If upgrading from any system version prior to the System 800xA 5.1 release, the MOD 300 Group Graphic display will continue to use the Visual Basic Group Graphic Element as default. This will remain until the Process Graphic 2 aspect of the same name is overridden. This will cause only this instance of the MOD 300 Group Graphic display to use the Process Graphic 2 aspect as default.

Dynamic Value

The MOD_DValue graphic provides a dynamic value graphic element to support MOD 300 data. The subelement is included in the Graphics Structure, Graphics Tools as an OCS Graphic Subelement. This provides the Graphic Builder the library support for OCS Graphic Subelements. In addition, pre-configured DValues for Measure, Output, Setpoint, Command and State are provided. When configuring graphic primitives, such as Advant Numeric, browse to a MOD object and then use the Expression Builder to add MOD OPC Control Connections and MOD 300 Alarm Summary connections. See Appendix B, Syntax for Database References.

Add a Graphic Aspect to an Object and Open Graphic Builder

1. Select the object requiring a graphic aspect in Plant Explorer and open the context menu.
2. Select New Aspect.
3. The New Aspect window is opened. Select Show all.
4. Select Graphic Element or Graphic Display from the dialog box, give the aspect a proper name and click Create.
5. Go to the aspect in the aspect list and select it. Open the context menu for the aspect and select Edit or click on Edit in the Preview Area.
6. The Graphic Builder opens up allowing the graphic aspect to be edited. Use the Graphics Builder to configure and deploy graphic aspects. See System 800xA Engineering - Process Graphics for information on using the Graphics Builder and for information on standard parameters.
Using MOD_DValue in Graphic Builder

The MOD_DValue is an OCS Graphic Subelement that includes specific parameters as described in Table 4. Build the graphic element using the data types suggested. Other standard parameters are: FillColor, Min, Max, TextAlignment, Visible, ToolTip, Font, FrameColor, FrameColor2, FrameWidth, TextColor, ScientificFormat and StandardFormat. ScientificFormat and StandardFormat use the standard Visual Basic formatting. Parameters not used are: ActivateCondition, AppErr, DataTypes, DefaultActivation, NumberOfDecimals, ShowMilliseconds, and StepSize.

ScientificFormat and StandardFormat parameters are the current default numeric display on the delivered MOD_DValues. To provide flexibility and to promote formats specific to objects or object groups, the MOD_DValue Format Definition aspect can be used. Refer to MOD_DValue Format Definition for Process Graphics 2 on page 92.

Table 4. Configuration Attributes for MOD_DValue

<table>
<thead>
<tr>
<th>MOD Parameters</th>
<th>Data Types Used (Property.Sub)(1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlarmSignal</td>
<td>_InAlarm.Value</td>
</tr>
<tr>
<td>SignalQuality</td>
<td>_InAlarm.QualityCode</td>
</tr>
<tr>
<td>AlarmString</td>
<td>_Alarmstr.Value</td>
</tr>
<tr>
<td>UnackSignal</td>
<td>_Unack.Value</td>
</tr>
<tr>
<td>EnableAlarms</td>
<td>True or False</td>
</tr>
</tbody>
</table>
Table 4. Configuration Attributes for MOD_DValue (Continued)

<table>
<thead>
<tr>
<th>MOD Parameters</th>
<th>Data Types Used (Property.Sub)(^{(1)})</th>
</tr>
</thead>
<tbody>
<tr>
<td>EnableInput</td>
<td>True or False</td>
</tr>
<tr>
<td>OPCStatus</td>
<td>Device Stat.QualityCode</td>
</tr>
<tr>
<td></td>
<td>Measure.QualityCode</td>
</tr>
</tbody>
</table>

\(^{(1)}\) *_ includes Measure, Output, Setpoint, Deviation and Device.

Fill Color is the background color if not in alarm.

Using Pre-configured DValues in Graphic Builder

Pre-configured DValues for Measure, Output, Setpoint, Command and State are available by browsing to an object, selecting Measure_DValue for example and then using the Add Element button to include the value on the graphic.

MOD_DValue Format Definition for Process Graphics 2

For Process Graphics 2 users, the MOD_DValue Format Definition aspect is created to allow for more flexible, real value representation on the MOD_DValue. It is a General Properties aspect with properties defined that are used by the MOD_DValue. By default, the MOD_DValue Format Definition aspect does not exist, and the MOD_DValue behavior will remain unchanged and use the input properties ScientificFormat and StandardFormat with hard-coded limits. If a MOD_DValue Format Definition aspect is created in the appropriate location, then it will make the ScientificFormat and StandardFormat input properties obsolete.


The MOD_DValue will find the aspect in one of the two ways. First, it will search using the input properties. If not found, it will search a hard-coded path which is [Graphics Structure]Graphics Tools/MOD 300 Graphics
**Subelements/MOD_DValue/MOD_DValue Format Definition.** This aspect does not exist unless created by copying and renaming the template. If neither is found, the MOD_DValue will fall back to the old behavior (similar to the previous Visual Basic version) and use the ScientificFormat and StandardFormat variables.

The two new (optional) input properties used to define where to search for this Format Definition aspect are as follows:

**FormatConstants AspectName (string):** The name of the MOD_DValue Format Definition aspect. A copy of the default MOD_DValue Format Definition Template aspect can be made and named as Measure Format Definition, for example. Set this value to be Measure Format Definition and this MOD_DValue will now have a unique Format Definition aspect. If this value is not specified or there is an error during lookup, it will default to the MOD_DValue Format Definition aspect in the Graphics Structure on the PG2 MOD_DValue object. **Default Value:** MOD_DValue Format Definition.

**FormatConstantsPath (string):** The path to the MOD_DValue Format Definition aspect. To search on the current object where this MOD_DValue aspect lives, this string should be “.”. Otherwise, it is necessary to specify a correct Aspect Directory path such as: [Object Type Structure]Object Types/Control System/MOD 300/MOD Control System/MOD_CCF_Group/MOD_CCF_PID_LOOP. If the specified path string is invalid, then the aspect located in the Graphics Structure on the MOD_DValue object type will be used. **Default Value:** “.”

This allows the flexibility of multiple formatting options. Some examples of other Use Case scenarios follow:

- A user might choose to copy the Format Definition Template to [Graphics Structure]Graphics Tools/MOD 300 Graphics Subelements/MOD_DValue/MOD_DValue Format Definition. This will cause all PG2 MOD_DValues in the system to use the formats defined in this aspect on the next callup. If the Format Definition is modified from this point on, the changes are live (no reopening of the displays is required). There is one location to manage the formats for all PG2 MOD_DValues.

- Alternatively, a Format Definition may be placed in the “.” path, allowing modifications to affect only a subset of the MOD_DValues in the system.
The input properties allow a new aspect name and path to be specified on a per-instance level for the MOD_DValue allowing finer grained control, if required. This requires editing graphics or elements to set these input properties.

An example of Format Definition is shown in Figure 36.
**TCL Array Plot**

The TCL Array Plot is configured using the MOD 300 Plot Aspect as shown in Figure 37.

---

**Create MOD 300 Plot Aspect**

1. Select an object (usually MOD_UNIT) and then add a new aspect (**New Aspect** from the context menu).
2. From the **New Aspect** dialog, enable the check box **Show All**.
3. In the list of common aspects select the **MOD 300 Plot Aspect** and give it a name.
4. Click **Create**.

---

*Figure 37. MOD 300 Plot Aspect Configuration*
Configure MOD 300 Plot Aspect

Two primary ways to configure a TCL array are: the Plot and the Filled Plot.

The Plot display shows data contained in a TCL array as a contiguous trace. As many as two traces of data can be plotted at one time.

The Filled Plot display is a single trace, filled with a color between the x-axis and the trace line.

TCL arrays can have vertical blue lines drawn at user-specified locations. Plots of this type are called Hybrid Plots. Hybrid plot displays are useful for flat sheet and similar applications. A TCL hybrid line array, must be created before the hybrid plot display is usable. Hybrid line array configuration is discussed later in this section.

Minimum Configuration Requirements

The minimum configuration requirements are:

- Default Plot Attributes (Use Array not checked):
  - Unit Name.
  - Array Name.
  - # of Points.
  - x-axis width.
  - y-axis top and bottom.

The look of the actual plot will include a red trace with a red border and frame on a gray background, 5 primary blue horizontal ticks and 5 primary red vertical ticks, the grid and other attributes are off.

- In Plot Array Attributes (Use Array Attributes must be checked):
  - Unit Name.
  - Array Name.

The look of the actual plot depends upon the plot attributes in the array.
## Configuration Fields

### PLOT

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use Array Attributes</td>
<td>When checked, the plot trace attributes (# of Points, Y-Axis Top, Y-Axis Bottom, X-Axis Width, X-Start Position) are contained in the first twenty elements of a plot array. See In Plot Array Option. When not checked, the plot trace attributes are to be configured using the Trace 1 and Trace 2 tabs. The trace fields in the dialog box are configurable. Plot data starts at the first array element.</td>
</tr>
<tr>
<td>X-Axis Visible</td>
<td>When checked, the x-axis is visible as determined by the X-Axis Color and X-Axis Pos on the Grid tab.</td>
</tr>
<tr>
<td>Back Color</td>
<td>Use the <strong>Color</strong> selection menu to set the background color of the plot grid.</td>
</tr>
<tr>
<td>Ruler Color</td>
<td>Use the <strong>Color</strong> selection menu to set the ruler color. The ruler is the vertical line that indicates where on the trace the runtime ruler readout values are taken.</td>
</tr>
<tr>
<td>Frame Color</td>
<td>Use the <strong>Color</strong> selection menu to set the frame color.</td>
</tr>
<tr>
<td>Filled Plot</td>
<td>When checked, the filled plot is used. Only Trace #1 is available for the filled plot.</td>
</tr>
<tr>
<td>Hybrid Plot</td>
<td>When checked, the hybrid plot is used. Enter the Unit Name and Array Name on the Plot tab.</td>
</tr>
<tr>
<td>Y-Axis Visible</td>
<td>When checked, the y-axis is visible with scale values as set on the Trace tabs.</td>
</tr>
<tr>
<td>Unit Name</td>
<td>Configured Tag name of TCL unit to be monitored for the Hybrid Plot.</td>
</tr>
<tr>
<td>Array Name</td>
<td>Use the standard array name format for the Hybrid Line Array name.</td>
</tr>
</tbody>
</table>
GRID

Grid Color  Use the **Color** selection menu to set the color for the grid that appears over the background rectangle.

X-Axis Color  Use the **Color** selection menu to specify the color of the x-axis line. X-Axis Pos determines position and X-Axis Visible determines if it is shown.

X-Axis Pos  This is a valid option if the plot attributes are contained in the plot array. The position of the x-axis is set to either the CENTER, TOP, or BOTTOM of the y-axis.

X-Axis Ticks  This field is used to specify the number of major ticks on the x-axis (2-99). Two are always shown.

Vertical Grid Visible  When checked, the vertical grid is visible. The number of grid lines matches the number of x-axis ticks.

Y-Axis Ticks  This field is used to specify the number of major ticks on the y-axis (2-99). Two are always shown.

Horizontal Grid Visible  When checked, the horizontal grid is visible. The number of grid lines matches the number of y-axis ticks.

TRACE 1 / TRACE 2  (Only Trace #1 is available for the filled plot.)

Unit Name  Configured Tag name of TCL unit to be monitored.

Array Name  Enter the TCL array name to be plotted. Plot displays require a TCL floating point array to store the plot’s attribute and/or plot point values. The array being created must be large enough to store values for all the points that need to be plotted.

# of Points  This field is used to specify the number of points to be plotted. # of Points + the X-Start Position = X-Axis Width. Specifying more points than available will cause a diagnostic error message and will not display any points. Specifying less than 25 points will produce a low resolution plot (the ruler will offset to the nearest point during selection).
<table>
<thead>
<tr>
<th>Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y-Axis Top</td>
<td>Used to specify the top value of the y-axis.</td>
</tr>
<tr>
<td>Y-Axis Bottom</td>
<td>Used to specify the bottom value of the y-axis.</td>
</tr>
<tr>
<td>X-Axis Width</td>
<td>This field is used to specify the width of the x-axis for the trace in points.</td>
</tr>
<tr>
<td>X-Start Position</td>
<td>This field is used to specify the point on the x-axis to start plotting the data pointed to by the respective trace plot array.</td>
</tr>
<tr>
<td>Trace Color</td>
<td>Use the <strong>Color</strong> selection menu to specify the color of the trace and scale values for a trace. The trace will be red if limits are left blank. Visibility is set by Y-Axis Visible.</td>
</tr>
<tr>
<td>Offset</td>
<td>An offset (index) value can be entered to start plotting from a particular element. For example, 1 starts at 1. To support the Plot Array option, 0 starts plotting from element 21 to bypass the user array attribute values.</td>
</tr>
<tr>
<td>Limit High</td>
<td>Used to specify the high limit value of the y-axis. The value above the limit is drawn in red rather than in the specified normal trace color. Note: Leaving this blank sets the limit to 0 and the trace is always red.</td>
</tr>
<tr>
<td>Limit Low</td>
<td>Used to specify the low limit value of the y-axis. The value below the limit is drawn in red rather than in the specified normal trace color. Note: Leaving this blank sets the limit to 0 and the trace is always red.</td>
</tr>
</tbody>
</table>

**In Plot Array Option (Use Array Attributes Checked)**

When using the in plot array option, the first 10 elements of the array are used as the attribute values of the plot display (the first 20 elements of the array are reserved for this purpose). Table 5 shows the proper order in which the attribute values of the display should be entered. All other elements of the array may contain the plot point...
values. The points are drawn on the array plot display consecutively from left to right. Therefore, it is important to enter the values for the plot points in the proper elements of the array. Use the TCL Editor to enter plot points.

Do not use the array attribute table when there is a mixed system of HP-UX and Windows nodes. The element 6 Normal plot (bar) color numbers used with Windows are incompatible with the HP-UX nodes.

Table 5. Locations of Plot Display Attribute Values in Arrays

<table>
<thead>
<tr>
<th>For Array Plot</th>
<th>Plot Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element 1</td>
<td>Maximum number of points to plot (bars in display)</td>
<td>The same value entered via the attribute menu when the plot display was created.</td>
</tr>
<tr>
<td>Element 2</td>
<td>Base line value (the value to plot around) for all plots (bars) of the display</td>
<td>A value that is either the midpoint between the high and low limit values, equal to the low limit value, or the target value for the point.</td>
</tr>
<tr>
<td>Element 3</td>
<td>Scale Value</td>
<td>The value represented by one vertical tic on the vertical axes of the plot display.</td>
</tr>
<tr>
<td>Element 4</td>
<td>High limit value</td>
<td>The value above which the plot (the bar) is drawn in red rather than in the specified normal plot (bar) color.</td>
</tr>
<tr>
<td>Element 5</td>
<td>Low limit value</td>
<td>The value below which the plot (the bar) is drawn in red rather than in the specified normal plot (bar) color.</td>
</tr>
</tbody>
</table>
Table 5. Locations of Plot Display Attribute Values in Arrays (Continued)

<table>
<thead>
<tr>
<th>For Array Plot</th>
<th>Plot Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element 6</td>
<td>Normal plot (bar) color</td>
<td>A number identifying the color for the plots (bars) of the plot display. See <em>Determining Color Number for Normal Plot Bar</em> on page 103 for a description on how to determine a color number. Black = 0 White = 16777215 Red = 255 Green = 65280 Blue = 16711680 Yellow = 65535 Orange = 4227327 Purple = 16711807 Cyan = 16776960</td>
</tr>
<tr>
<td>Element 7</td>
<td>Redraw Flag</td>
<td>A value that triggers a refresh of the plot display axes. Upon refresh, the plot display redraws if this value has changed since the last refresh.</td>
</tr>
<tr>
<td>Element 8</td>
<td>Actual number of points to plot</td>
<td>The number of points (bars) to plot (subset of VALUE1 value). The remainder of the plot display has no bars.</td>
</tr>
<tr>
<td>Element 9</td>
<td>Offset of first data element</td>
<td>The plot display normally starts plotting from element 21. An offset value can be entered to start plotting from an element relative to element 21. Enter any positive number. Enter 0 as a default to start plotting from element 21.</td>
</tr>
<tr>
<td>Element 10</td>
<td></td>
<td>(Spare element—reserved for future use.)</td>
</tr>
</tbody>
</table>
Hybrid Line Array Option

If a hybrid plot is being created, then make a hybrid line array. The hybrid line array is a TCL real number array, and must be 260 elements long. The maximum number of vertical lines that can be configured is 256. The plot points and vertical lines are lines that are drawn on the plot display consecutively from left to right at runtime; the values for the plot points and vertical lines must be entered into the array elements accordingly. The format of the hybrid line array is given in Table 6.

Table 6. Hybrid Plot Display Attribute Values in Hybrid Line Arrays

<table>
<thead>
<tr>
<th>For Array</th>
<th>Plot Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element 1</td>
<td>Number of lines</td>
<td>Number of vertical lines to draw, maximum of 256.</td>
</tr>
<tr>
<td>Element 2</td>
<td>Vertical line color</td>
<td>A number from the following list, identifying the color for the vertical lines of the plot display:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Black</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Yellow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = Red</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = Green</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 = Slate</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 = Blue</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 = White</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 = Magenta</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9 = Orange</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10 = Cyan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>11 = Flashing Green</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12 = Flashing Cyan</td>
</tr>
<tr>
<td></td>
<td></td>
<td>13 = Flashing White</td>
</tr>
<tr>
<td></td>
<td></td>
<td>14 = Flashing Yellow</td>
</tr>
<tr>
<td></td>
<td></td>
<td>15 = Flashing Red</td>
</tr>
</tbody>
</table>
Table 6. Hybrid Plot Display Attribute Values in Hybrid Line Arrays (Continued)

<table>
<thead>
<tr>
<th>For Array</th>
<th>Plot Attribute</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Element 3</td>
<td>Vertical line texture</td>
<td>A number from the following list, identifying the texture for the vertical lines of the plot display:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>0 = Invisible</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1 = Solid</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2 = Dotted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3 = Dashed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4 = Dashed / Dotted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5 = Double Dashed / Dotted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6 = Long Dashed / Dotted</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7 = Double Dot / Dashed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>8 = Close Dash</td>
</tr>
<tr>
<td></td>
<td></td>
<td>9 = Double Dot / Long Dash</td>
</tr>
<tr>
<td>Element 4 through element 260</td>
<td>Vertical line placement of the 256 lines</td>
<td>Expressed as a percentage to place each vertical line, from left to right, across the plot display.</td>
</tr>
</tbody>
</table>

Determining Color Number for Normal Plot Bar

The color number entered for Element 6 in Table 5 is determined as follows:

\[
\text{Color Number} = \text{Red} + (256 \times \text{Green}) + (65536 \times \text{Blue})
\]

where the variables Red, Green and Blue can be determined from the Windows Desktop Display Properties menu, Appearance tab, Color (Other...) pull down selection to get the Color selection display as shown in Figure 38.
For example, the value for Yellow is calculated as follows:

\[
\text{Yellow Color Number} = 255 + (256 \times 255) + (65536 \times 0)
\]

\[
\text{Yellow Color Number} = 65535
\]

**MOD 300 Group Trend Template**

MOD 300 trend colors and columns are found in the Trend Template aspect (Library Structure/Trend Templates/MOD 300 Group Trend Template). Trace colors, and trend display colors are set on the Colors tab. Columns are set on the Columns tab.
MOD 300 Trend Buffer

Any MOD 300 tag that requires controller trend data from the MOD 300 trend buffer needs a Log Configuration aspect as shown in Figure 39. This allows group trends and trend aspects to present the most recent 300 data points from the controller trend buffer without having to go through the history logs. Select a trend template that matches the trend rate of the loop (6, 12, 60, 120, 360, 720).

Figure 39. Adding Log Configuration Aspect to MOD 300 Tag
MOD 300 Alarm Colors Aspect

The default MOD 300 Alarm Colors are defined as an aspect of the System Workplace, Workplace Structure. The aspect can also be found under the Plant Explorer Workplace or Operator Workplace, Workplace Structure. Changes made to the root structure are inherited by the workplaces.

1. Select the Web System Workplace object in the Workplace Structure.
2. Select the MOD 300 Alarm Colors aspect.
3. Select a Logical Color Name to modify the color as shown in Figure 40.

4. Select the RGB Value and click Modify to set the value.

Figure 40. MOD 300 Alarm Colors, Logical Color Definition Dialog
5. Click **Apply** to save the logical color.

Alarm colors for the Loop Detail display use the Acknowledge / Background color combination. Loop Detail color combinations are fixed.

Faceplate displays use the Acknowledge / Unacknowledged color combination.

**MOD 300 Display Colors**

Color settings are determined by the Windows settings on the Display Properties dialog Appearance tab. Color configuration parameters for the Group and Area displays can be set to further modify the Windows settings. The following registry parameters, if present, will modify the Windows settings. The location is:

```
HKEY_LOCAL_MACHINE > SOFTWARE > Wow6432Node > ABB > MOD Add-on > config
```

The registry values are:

- ChangeableColorAreaGroupDisplay.
- LimitColorAreaGroupDisplay.
- StaticColorAreaGroupDisplay.

These are REG_DWORD parameters whose values can be set using the value pattern of 0x00bbggrr where bb is for blue, gg is for green and rr is for red values from 00 to FF.

When using the registry keys to set text colors for group and area displays, the text colors do not reverse if the color behind the text is the same as the text.

The decimal form of the variables *Red, Green* and *Blue* can be determined from the Windows Desktop Display Properties dialog, Appearance tab, Advanced Appearance, Color (Other...) pull down selection to get the Color selection display. For example,

- Red is converted to bb=00, gg=00, rr=FF
- Green is converted to bb=00, gg=FF, rr=00 and
- Blue is converted to bb=FF, gg=00, rr=00

To closely match the Multibus display colors, set:
MOD 300 NLS Resources

Native Language Support (NLS) resources are available in the Object Type Structure for MOD 300 Displays.

Select the NLS Resource Manager aspect within the structure:

Object Types / Control System / MOD 300.

From the General tab select a Resource ID and then use the Translated text area to modify the text.

The resource IDs listed in Table 7 are used in the displays and care must be taken to match the space given.

Table 7. NLSID Resource Usage

<table>
<thead>
<tr>
<th>Resource ID</th>
<th>Text</th>
<th>Where Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLSID_Auto</td>
<td>Auto Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_AUTO</td>
<td>AUTO</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Bi</td>
<td>Bi</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Bi_Value</td>
<td>Bias Value</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_COMP</td>
<td>COMP</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_COMP_MODE</td>
<td>Computer Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_DevRest</td>
<td>RSTON</td>
<td></td>
</tr>
<tr>
<td>NLSID_DEVSTATE</td>
<td>STATE</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_FB</td>
<td>FB</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_FB_Lowered</td>
<td>Move loop out of FeedBack mode</td>
<td>Faceplate tool tip.</td>
</tr>
</tbody>
</table>
### Table 7. NLSID Resource Usage (Continued)

<table>
<thead>
<tr>
<th>Resource ID</th>
<th>Text</th>
<th>Where Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLSID_FB_Raised</td>
<td>Move loop into FeedBack mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_FF</td>
<td>FF</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_FF_Lowered</td>
<td>Move loop out of FeedForward mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_FF_Raised</td>
<td>Move loop into FeedForward mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_FFFB</td>
<td>FF/FB</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_FFFB_Lowered</td>
<td>Move loop out of FeedForward/FeedBack mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_FFFB_Raised</td>
<td>Move loop into FeedForward/FeedBack mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_LOC</td>
<td>LOC</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_LOC_BI</td>
<td>Local Bias Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_LOC_Mode</td>
<td>Local Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_LOC_RA</td>
<td>Local Ratio Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_LOC_SP</td>
<td>Local Setpoint Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_Local_BI</td>
<td>L-BI</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Local_RA</td>
<td>L-RA</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Local_SP</td>
<td>LOC</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_MAN</td>
<td>MAN</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_ModSetState</td>
<td>Set to State</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_Next</td>
<td>Next</td>
<td>Faceplate</td>
</tr>
<tr>
<td>NLSID_NextPrevious</td>
<td>Display Next/Previous States</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_Out</td>
<td>Out</td>
<td>Faceplate.</td>
</tr>
</tbody>
</table>
### Table 7. NLSID Resource Usage (Continued)

<table>
<thead>
<tr>
<th>Resource ID</th>
<th>Text</th>
<th>Where Used</th>
</tr>
</thead>
<tbody>
<tr>
<td>NLSID_Out_value</td>
<td>Faceplate.</td>
<td></td>
</tr>
<tr>
<td>NLSID_Output_value</td>
<td>Output value</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_Previous</td>
<td>Previous</td>
<td>Faceplate</td>
</tr>
<tr>
<td>NLSID_Process_value</td>
<td>Process value</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_Pv</td>
<td>Pv</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Ra</td>
<td>Ra</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_RA_Value</td>
<td>Ratio Value</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_REM_BI</td>
<td>Remote Bias Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_REM_RA</td>
<td>Remote Ratio Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_REM_SP</td>
<td>Remote Setpoint Mode</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_Remote_BI</td>
<td>R-BI</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Remote_RA</td>
<td>R-RA</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Remote_SP</td>
<td>REM</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_R-LOC</td>
<td>LOC</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_R-REM</td>
<td>REM</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_Setpoint_value</td>
<td>Setpoint value</td>
<td>Faceplate tool tip.</td>
</tr>
<tr>
<td>NLSID_Sp</td>
<td>Sp</td>
<td>Faceplate.</td>
</tr>
<tr>
<td>NLSID_SubstitutedQualitySymbol</td>
<td>&amp;</td>
<td>Faceplate.</td>
</tr>
</tbody>
</table>
Appendix A  Advant OCS Database

Requirements

800xA for MOD 300 requires a Connectivity Server (with RTAB) to connect to the Distributed Communications Network (DCN).

A server that provides access to controllers and other sources for real-time data, historical data, and alarm and event data. A Connectivity Server runs services related to OPC/DA, OPC/AE, OPC/HDA. Several groups of connectivity servers may exist in a system, each serving one set of data sources, for example, MOD 300 related controllers and AC 800M Controllers.

To properly represent the Connectivity Server node in the MOD 300 database, a GENERICD object must be configured in the MOD 300 database. Create one for each Connectivity Server. If a node is being replaced, use either a GENERICD or CONSOLE node. The GENERICD object is a child of the AREA object and represents a subsystem in the area. To route message logs to 800xA for MOD 300, use the message routing configuration fields in the Area template.

If parameter change messages (CCF_EVENT, TCL_EVENT) are needed, a Logger template is required under the GENERICD and must have the routing changed to it. The same is true if user log on/off or user acknowledge messages are needed. These are both needed to store this information into Information Management.

The GENERICD and AREA database objects are inserted into the MOD 300 database hierarchy and object attributes are defined in the following sections and as described in the following:


Appendix A  Advant OCS Database

The Template Builder view of the GENERICD object is shown in Figure 41.

Figure 41. GENERICD Object, Template Builder Window

The GENERICD object attributes are described below.
This is the DCN node address. This must be the same DCN address that was defined for the MOD 300 Connectivity Server node during software loading. The address is expressed as a decimal number. Any number from 1 to 255 is valid; however, addresses with a lower order hexadecimal value of one are reserved for configurator/data processor (CDP) nodes and SHOULD NOT BE USED for the MOD 300 node, Table 8. Be sure to enter a decimal number in the PHYSICAL DEVICE field.

Table 8. DCN Addresses Reserved for Nodes with Configurator Software

<table>
<thead>
<tr>
<th>CDP Reserved Address (Hex)</th>
<th>Decimal Equivalent</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>1</td>
</tr>
<tr>
<td>11</td>
<td>17</td>
</tr>
<tr>
<td>21</td>
<td>33</td>
</tr>
<tr>
<td>31</td>
<td>49</td>
</tr>
<tr>
<td>41</td>
<td>65</td>
</tr>
<tr>
<td>51</td>
<td>81</td>
</tr>
<tr>
<td>61</td>
<td>97</td>
</tr>
<tr>
<td>71</td>
<td>113</td>
</tr>
<tr>
<td>81</td>
<td>129</td>
</tr>
<tr>
<td>91</td>
<td>145</td>
</tr>
<tr>
<td>A1</td>
<td>161</td>
</tr>
<tr>
<td>B1</td>
<td>177</td>
</tr>
<tr>
<td>C1</td>
<td>193</td>
</tr>
<tr>
<td>D1</td>
<td>209</td>
</tr>
<tr>
<td>E1</td>
<td>225</td>
</tr>
<tr>
<td>F1</td>
<td>241</td>
</tr>
</tbody>
</table>
AUTO START

This field determines whether or not the node starts automatically when the software is downloaded. The only valid choice for a GENERICD node is YES.

SOFTWARE NAME

This field determines the functionality of a Multibus-based GENERICD node. This field is not applicable for the node. Leave it at its default value.

DO RATE

This attribute is not applicable for the 800xA for MOD 300 application.

SECONDARY DP, BACKUP ENABLE, and BACKUP OVERRIDE

These fields have not been implemented. Leave them at their default values.

DEFAULT ENVIRONMENT

This attribute is not applicable for the 800xA for MOD 300 applications.

NODE TYPE

This field specifies the hardware platform for the GENERICD node. The choices are:

- MOD300_NODE This option is for a Multibus-based node.
- ADVANT_STATN This option is for a node with 800xA MOD 300 software or any Advant OCS workstation.
AREA Object

Message logs to be routed to 800xA for MOD 300 use the message routing configuration fields in the Area template. Scroll to display the MESSAGE CENTER fields (AREA EDITORS fields are not applicable), Figure 42 (representative). Set up these structures for history message logs. If this has not been done already, then recompile and install.

Figure 42. Message Center Fields for CONSLIB or Area Object
Message Center Fields

Message fields are used to specify the destinations of messages originating in this configuration area. During runtime, the following types of messages can be generated by the system: CCF event, CCF alarm, TCL event, TCL billboard and system error (diagnostic).

Only the messages that are absolutely necessary should be configured. The list of message destinations set up in the system database has a direct bearing on the communications load. Each destination for a message adds to the load.

The destination and type of messages are defined by entries to the following fields:

MESSAGE CTR Use an object ID for the device to receive messages. The ID must be the one assigned when the object was inserted in the database.

MESSAGE TYPE Is the type of message. Valid entries for type are:

- CCF_ALARM
- CCF_EVENT
- CCF_BOTH - This option designates both CCF alarm messages and event messages.
- TCL_EVENT
- TCL_BILLBOARD - This option is used to designate TCL unit messages that are generated by TCL MESG and REPLY statements and programmable unit alarms generated by TCL UNIT_ALARM statements.
- TCL_BOTH - This option designates both TCL event messages and billboard messages.
- SYSTEM
- BOTH - This option designates all message types designated by CCF_BOTH and TCL_BOTH.
- ALL - This option designates all message types.

REMOTE TYPE The entry specifies the object type. Leave the field blank unless using multiple CDP's.
Message Center Entries for Loggers

Loggers are software devices that log messages to printers and/or disks. An entry must be made in the MESSAGE CTR and MESSAGE TYPE fields for each type of message to be logged on a specific logger. Any combination of message types is valid. If a system has more than one configuration area, entries must be made to the MESSAGE CTR and MESSAGE TYPE fields of each AREA object.

In the following examples, **LOGGER1** is the object ID assigned to the object representing this Logger.

**Example 1** - These entries send all CCF and SYSTEM messages to LOGGER1:

```
LOGGER1 CCF_BOTH
LOGGER1 SYSTEM
```

**Example 2** - This entry sends all CCF and TCL messages to LOGGER1.

```
LOGGER1 BOTH
```

**Example 3** - This entry sends all messages to LOGGER1.

```
LOGGER1 ALL
```

Message Routing from Operator Station

To route messages, enter the message destinations and the types of messages to be routed in a MSG_ROUT object to the destinations in the MESSAGE CTR edit window of the CONSLIB object. This object provides the means to configure how console log on and/or alarm acknowledgment messages are routed.

Indicate the message destination by entering the object ID of the destination package. For message type enter any one of the following:

- **CONS_LOG_ONS**    Console log on
- **CONS_ACKS**        Console alarm acknowledgment
- **CONS_BOTH**        Both console log on and console alarm acknowledgment

There is a limit of 256 destinations for messages. Messages of a specific type from a configuration area can be sent to no more than 40 destinations.
Recommended Message Centers for Routing

Route to the node: CCF_ALARM, TCL_BILLBOARD, SYSTEM
Route to Logger: CCF_EVENT, TCL_EVENT, CONS_ACK, CONS_LOG

Changes to Loop Type

Whenever the loop type information changes (such as when adding or deleting Auto/manual FCM’s, PID’s or adaptive functions to a PID), it causes OMF Object ID’s to change and not be picked up because the Object ID already exists.

To maintain the OMF object resolution, some maintenance on the nb.store file located in the install path must be performed. The default is:

\C:\ProgramData\ABB\SystemServices\OMF.

Specifically, delete the object from the system and on every Connectivity Server, delete the nb.store file and then reboot the node so the file can be rebuilt.

Memory Pool Utility (mputil)

The mputil utility is intended to be used by knowledgeable ABB personnel or customers. The utility has no specific knowledge of how pools should be configured to allow a functioning system. The utility makes it possible to change memory pool allocations so that the RTAB node will not start. For assistance, contact a field support representative.

The mputil utility is used to display and change the allocation of memory to the MTOS memory pools on the Run Time Accelerator Board (RTAB). Changes take effect the next time the RTAB or Connectivity Server is restarted. It works by modifying the MTOS memory pools table overlay file, pu511_pools.x, used with a PCI RTAB. The file is loaded into the RTAB during its start-up sequence. The most common use of mputil is to enlarge the BACKMEM pool for large customer applications. CCF, TCL and DBMS are examples of system applications that use BACKMEM.

Navigate to the directory path of the program to run it (C:\Program Files(x86)\ABB 800xA\SystemServices\MODBase\bin). The MTOS Memory Pool Allocation Utility program is menu-driven.
To change allocations, write permission is required for the `pu511_pools.x` file in the `pu511load` directory. Also, be sure to know how much RAM is on the RTAB because the utility will ask for it. The System Status page is one source of this information.

When the `mputil` is run, the following menu of options are presented:

```
MTOS Memory Pool Allocation Utility Ver 1.1
D: Current RTA Node Configuration
Q: Quit
Select one :
```

Select `D` to enter the RTA node configuration. Enter the amount of RTAB memory for the PU511 at the next prompt. The current/custom memory configuration, and the original default configuration, are displayed. Below is a representation of the main `mputil` display where memory allocation is done.

![MTOS Memory Pool Allocation Utility](image)

*Figure 43. MTOS Memory Pool Allocation Utility*
The custom configurations can be modified. The first time the utility is run, the default configuration is saved to the file:

pu511_pools.default

Changes to the custom file will not be made until they are saved. Therefore, feel free to experiment until the right configuration is achieved.

To save the custom pool values, select S. The opportunity to save the old pools overlay file to any filename is provided. Write permission is required in the pu511load directory. The new changes are saved back to the pu511_pools.x file, which is the file loaded onto the RTAB.

There are other files in the pu511load directory. Do not use any files that you have not created.

To change the amount of memory to configure on the RTAB, select M, then select 2 for 16 Mb (PCI RTABS have 16 Mb of memory).
Appendix B  Syntax for Database References

Format of Expressions

This appendix describes the syntax of database references to process objects. These references are used in the expression builder as described in the *System 800xA Engineering Process Graphics (3BSE049230*) or *System 800xA Engineering Process Graphics based on Visual Basic (3BSE030335*) instructions. Database references to the following types of parameters (object properties) are supported:

- Loop parameters.
- Loop FCM parameters.
- Array element parameters.
- Recipe item parameters.
- Taylor Ladder Logic (TLL) parameters.

The set of available parameters that can be used in the expression builder is determined by what objects were imported. In addition to the standard Alarm List, Alarm Global Properties and Name type parameters provided by the base software, MOD Connect includes MOD OPC Control Connection, and MOD300 Alarm Summary type parameters for connection.

Figure 44, shows the expression builder with a MOD_DValue configured using a PID loop. The property values shown for the loop are typical.
Loop Parameter Reference (Tag Access) Format

Loop parameter references are used when referencing a device or continuous loop parameter that is unique in the loop. For instance, a continuous control loop can have only one measured variable (MEASURE) and only one FCM having a SETPOINT (PID controller).

Two elements are necessary to reference this kind of value: the name of the loop (the tag), and the parameter’s mnemonic. Loop accessible parameter mnemonics are described in 3BUR000239Rnn01, Data Base Tables.
When creating an Advant Numeric, the loop parameter is referenced as:

\[ \text{tag:opc:parameter} \]

where

- **tag** is the name (tag) of the loop defined on the Loop Definition Template or Device Loops Template. This must be imported to be able to browse to the object.
- **opc** If the connection is through the OPC DA server then this will be MOD OPC Control Connection. If it is through the OPC AE server then it is MOD300 Alarm Summary.
- **parameter** is the parameter mnemonic

In addition, the sub parameter type must also be selected (for example Value).

For example:

- CV101:MOD OPC Control Connection:DEV_CMND
- F125:MOD OPC Control Connection:MEASURE
- FIC105:MOD300 Alarm Summary:MEASURE_ALARMSTR

**FCM Parameter Reference (Tag-FCM Access) Format**

An FCM parameter reference is used when referencing a parameter that can occur in more than one FCM of a continuous control loop. For instance, a loop can have an input source in more than one of its FCMs, making it necessary to identify the FCM for the desired value, as well as the loop’s tag.

The concept of FCMs is explained in 3BUR00238Rnn01, *Configurable Control Functions (CCF) User’s Guide*. A knowledge of the FCMs in the system is necessary when using this type of database reference.

Three elements are required to reference this kind of value: the name of the loop (the tag), the name of the appropriate FCM, and the parameter’s mnemonic. FCM parameter mnemonics are described in 3BUR00239Rnn01, *Data Base Tables*. 
When creating a Advant Numeric, the loop parameter is referenced as:

\[ \text{tag:opc:fcmname/parameter} \]

where

- \textit{tag} is the name (tag) of the loop defined on the Loop Definition Template or Device Loops Template. This must be imported to be able to browse to the object.
- \textit{opc} If the connection is through the OPC DA server then this will be MOD OPC Control Connection. If it is through the OPC AE server then it is MOD300 Alarm Summary.
- \textit{fcmname} is the name of the FCM defined on the Loop Definition Template
- \textit{parameter} is the parameter mnemonic

In addition, the sub parameter type must also be selected (for example Value). For example:

- F125:MOD OPC Control Connection:LKP/RESULT
- TIC_10:MOD OPC Control Connection:A\text{IN/RESULT

### Array Element Parameter Reference Format

Array element parameter references are used when referencing a particular element of a TCL array.

Four items are necessary to reference this kind of value: the name of the unit where the TCL array is loaded, the array’s name, the location of the element within the array, and the type of data stored in the array.

The format for the array element parameter is:

\[ \text{unitname:array/arrayname/index1/index2/parameter} \]

where:

- \textit{unitname} is the name of unit where the array was loaded by a TCL statement.
Recipe Item Parameter Reference Format

Recipe item parameter references are used when referencing a particular recipe item. Four elements are necessary to reference this kind of value: the name of the unit where the recipe is loaded, the recipe item, the type of data the item contains, and the parameter. The format for the recipe item parameter reference is:

```
array arrayname index1 index2 parameter
```

- `array` is a keyword that defines the reference format.
- `arrayname` is the name of the array whose element value will be displayed.
- `index1` is the location within the first dimension of the array, expressed as an integer between 0 and 65536, for the element whose value will be displayed.
- `index2` is the location within the second dimension of the array, expressed as an integer between 0 and 65536, for the element whose value will be displayed. This location is only required when the array is two dimensional.
- `parameter` is the mnemonic for the value to display. The only valid parameter for all data types (REAL, STRING, INTEGER, BYTES, and TWO_BYTES) is:

  `VAL` (value of item).

The data type is automatically determined from the array data.

In addition, the sub parameter type must also be selected (for example Value).

For example:

```
UNIT1DPSS:array/IARRAY/6/VAL
```

The location specified for the element to be displayed must be within the array’s declared size. For example, to display the value for an element of a single dimensional array containing 10 elements, enter an integer from 1 to 10 for `index1`. Entering a location larger than the number of elements in the array will result in a bad quality error or graphic error on the runtime display instead of the element’s value. This is because there is no such location within the array from which to retrieve a value.
A Taylor Ladder Logic (TLL) parameter reference is used when referencing a parameter of a TLL element.

Three components are required to reference this kind of value: the name of the TLL element, the name of the TLL type, and the TLL parameter’s mnemonic.
The format for the TLL parameter reference is:

\[\text{tag:opc:type/parameter}\]

where

- **tag** is the name of the TLL element defined in the TLL Template.
- **opc** If the connection is through the OPC DA server then this will be MOD OPC Control Connection. If it is through the OPC AE server then it is MOD300 Alarm Summary.
- **type** is the name of the TLL element type as follows:
  - I_O Input/Output Points
  - TIM Timer
  - CNT Counter
  - REG Register
- **parameter** is the TLL parameter mnemonic. TLL parameter mnemonics are described in 3BUR000243Rnn01, Taylor Ladder Logic (TLL).

In addition, the sub parameter type must also be selected (for example Value).

For example:

- COUNT1:MOD OPC Control Connection:CNT/AC
- TIMER1:MOD OPC Control Connection:TIM/AC
Appendix C  Batch Support

Introduction

The MOD_PHASE object is the 800xA for MOD 300 interface to 800xA Batch Management and is equivalent to an 800xA Batch Equipment Phase. It provides access to a valid MOD TCL Sequence that has been adapted to support 800xA Batch ISA S88 compliant Phase processing.

This appendix describes how to use the S88Phase.tcl template to customize sequences that will be used as batch phases. This is required to translate TCL States, Abnormal, and Commands into S88 Complaint States, Error, and Commands. In addition, MOD_PHASE aliases are described and cross reference tables between S88 and TCL are included.

Usage

S88 Phases can be re-used across several MOD_UNITs. While they can be dynamically loaded into the MOD 300 controller using 800xA Batch Management, sequences should be auto-loaded into the controller whenever possible to minimize controller and communications loading caused by dynamic loading and removing and to reduce the potential for memory fragmentation in the controller.

Creating a Sequence for Batch Phases

Use the TCL Builder to customize a MOD Sequence that will be used as a batch phase. The S88PHASE.TCL file is used as a template to create the custom batch phase and can be found in the following directory:

```
Program Files(x86)\ABB 800xA\MOD 300\Batch\TclTemplate\S88PHASE.TCL
```
Copy this TCL file to the AdvaBuild machine. TCL Sequence names must be unique within first 8 characters and can be a total of 12 characters.

There are three main parts to the template:

- The header includes UserArray declarations and the Change State subroutine.
- Abort/Stop/Hold/Pause Phase Subroutines.
- Application Steps.

The fixed TCL code embedded within S88Phase.tcl implements a translation of TCL Modes and States to S88 Phase states. It utilizes TCL Abnormal routines to implement S88 States and TCL Step routines for the S88 Running State. The use of a TCL UnitArray allows for multiple instances of the S88Phase.tcl to run in parallel on a unit and resolves the TCL operational state into an S88 State.

While TCL Abnormal routines are required to implement S88 States, the use of custom user abnormals in Batch Phases is not recommended. Instead, use separate TCL sequences for monitoring applications. Refer to Program Files(x86)\ABB 800xA\MOD 300\Batch\TclTemplate\USERABN.TCL.

Edit the file as follows:

1. Add the alias identifier. This is of the type Integer and must be the first parameter in the parameter list. Any additional parameters are added after the ALIASID.

   The ALIASID parameter in the TCL PROC statement is set from the AliasId defined in the General Properties aspect of the MOD_PHASE object.

2. Declare the following local and data base variables.

   - **VAR**
     
     PAUSEREQUEST : INTEGER;

   - **DBVAR**
     
     SSTATUS : INTEGER;

3. Create Unit Array. The UNITVAR definition is required, change S88PHASE to myphase where myphase is your phase name. The maximum length of this name is 12 characters including the underscore (_). If your phase name is greater than 11 characters, it will need to be truncated. Be aware that the unit array names must unique within a specific unit.
4. Add Change State subroutine using `myphase` in place of S88PHASE for each case statement.

   1: `myphase_[1] := STATE;`
   ...
   9: `myphase_[9] := STATE;`

5. Copy these standard batch subroutines:

   ABORTPHASE
   STOPPHASE
   HOLDPHASE
   PAUSEPHASE

Leave the WAIT statements in place where there are no batch subroutines defined. TCL Abnormal status 3, 9, 11, and 13 are reserved for these abnormal subroutines. TCL Abnormal status 1, 2, 4 and 10 are used for the existing system abnormal conditions.

6. In the first step of your sequence, copy these Abnormal subroutine definitions for the above subroutines.

   ABNORMSUBR(ABNORM(3),ABORTPHASE);
   ABNORMSUBR(ABNORM(9),STOPPHASE);
   ABNORMSUBR(ABNORM(11),HOLDPHASE);
   ABNORMSUBR(ABNORM(13),PAUSEPHASE);

7. Copy step 'Complete' logic. This must be the last step of the sequence.

After the file is edited, it can be compiled for download to the controller. The download can be handled by an autoload sequence or by the 800xA Batch Manager software.
MOD_Phase Aliases

When a phase is executed in parallel on a unit, then aliasing must be used. Batch Management manages the alias ID to handle parallel instances of the same Phase in Recipes. A simple example is shown in Figure 45.

1. Set the **Source Name** of the XWAIT, MOD_PHASE object on the General Properties aspect to XWAIT, Figure 46. This is the name of the TCL sequence.

2. The **AliasID** of XWAIT is equal to the integer 1 by default.

3. Set the **Source Name** of the XWAIT_1, MOD_PHASE object on the General Properties aspect to XWAIT. XWAIT_1 identifies a unique object name but the source name of the TCL sequence remains the same, XWAIT.

4. Set the **AliasID** of XWAIT_1 equal to the integer 2. This allows a second XWAIT to be run parallel on the unit.

Up to nine aliases can exist on a unit. The index is not used to determine an order.

Aliased phases can not be loaded into the controller either by hand or by an auto-load sequence. Only 800xA Batch Management can load aliased sequences.

Figure 45. Parallel XWAIT Phases
Figure 46. Defining an Alias for the XWAIT and XWAIT_I Phases

S88 Procedure Commands and States for TCL Sequences

Table 9 describes the relationship between S88 Procedure commands and TCL Sequence commands. Table 10 describes the relationship between S88 Procedure states and TCL Sequence states. The matrix for valid Batch Management commands per state and state transition can be found in the batch operation manual. The TCL
Sequence state transitions are described in the Taylor Control Language (TCL) User Guide.

Table 9. S88 Procedure Commands for TCL Sequences

<table>
<thead>
<tr>
<th>S88 Procedure State</th>
<th>TCL Sequence State</th>
<th>TCL Abnormal Status</th>
<th>S88 Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>START</td>
<td>ACTIVE</td>
<td>NORMAL</td>
<td>This command orders the procedural element to begin executing the normal RUNNING logic. This command is only valid when the procedural element is in the IDLE state.</td>
</tr>
<tr>
<td>STOP</td>
<td>No Change</td>
<td>ABNORMAL 9</td>
<td>This command orders the procedural element to execute the STOPPING logic. This command is valid when the procedural element is in the RUNNING, PAUSING, PAUSED, HOLDING, HELD, OR RESTARTING state.</td>
</tr>
<tr>
<td>HOLD</td>
<td>No Change</td>
<td>ABNORMAL 11</td>
<td>This command orders the procedural element to execute the HOLDING logic. This command is valid when the procedural element is in the RUNNING, PAUSING, PAUSED or RESTARTING state.</td>
</tr>
<tr>
<td>RESTART</td>
<td>RESUME</td>
<td>N/A</td>
<td>This command orders the procedural element to execute the RESTARTING logic to safely return to the RUNNING state. This command is only valid when the procedural element is in the HELD state.</td>
</tr>
<tr>
<td>ABORT</td>
<td>No Change</td>
<td>ABNORMAL 3</td>
<td>This command orders the procedural element to execute the ABORTING logic. The command is valid in every state except for IDLE, COMPLETED, ABORTING and ABORTED.</td>
</tr>
<tr>
<td>RESET (viewed as Start at Beginning)</td>
<td>No Change</td>
<td>NORMAL</td>
<td>This command causes a transition to the IDLE state. It is valid from the COMPLETE, ABORTED, and STOPPED states.</td>
</tr>
</tbody>
</table>
Table 9. S88 Procedure Commands for TCL Sequences (Continued)

<table>
<thead>
<tr>
<th>S88 Procedure State</th>
<th>TCL Sequence State</th>
<th>TCL Abnormal Status</th>
<th>S88 Command Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAUSE</td>
<td>No Change</td>
<td>ABNORMAL 13</td>
<td>This command orders the procedural element to pause at the next programmed pause transition within its sequencing logic and await a RESUME command before proceeding. This command is only valid in the RUNNING state.</td>
</tr>
<tr>
<td>RESUME</td>
<td>RESUME</td>
<td>No Change</td>
<td>This command orders a procedural element that has PAUSED at a programmed transition as the result of either a PAUSE command or a SINGLE STEP mode to resume execution. This command is only valid when the procedural element is in the PAUSED state.</td>
</tr>
<tr>
<td>Continue</td>
<td>RESUME</td>
<td>No Change</td>
<td>This command resumes execution fromPaused State also.</td>
</tr>
</tbody>
</table>

Table 10. S88 Procedure States for TCL Sequences

<table>
<thead>
<tr>
<th>S88 Procedure State</th>
<th>TCL Sequence State</th>
<th>TCL Abnormal Status</th>
<th>S88 State Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IDLE</td>
<td>INACTIVE</td>
<td>NORMAL</td>
<td>The procedural element is waiting for a START command that will cause a transition to the RUNNING state.</td>
</tr>
<tr>
<td>RUNNING</td>
<td>ACTIVE</td>
<td>NORMAL</td>
<td>Normal operation.</td>
</tr>
<tr>
<td>COMPLETE</td>
<td>INACTIVE</td>
<td>NORMAL</td>
<td>Normal operation has run to completion. The procedural element is now waiting for a RESET command that will cause a transition to IDLE.</td>
</tr>
</tbody>
</table>
Table 10. S88 Procedure States for TCL Sequences (Continued)

<table>
<thead>
<tr>
<th>S88 Procedure State</th>
<th>TCL Sequence State</th>
<th>TCL Abnormal Status</th>
<th>S88 State Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>PAUSING</td>
<td>ACTIVE</td>
<td>ABNORMAL 13</td>
<td>The procedural element or equipment entity has received a PAUSE command. This will cause the procedural element to stop at the next defined safe or stable stop location in its normal RUNNING logic. Once stopped, the state automatically transitions to PAUSED.</td>
</tr>
<tr>
<td>PAUSED</td>
<td>PAUSED</td>
<td>NORMAL</td>
<td>Once the procedural element has paused at the defined stop location, the state changes to PAUSED. This state is usually used for short-term stops. A PAUSE command causes transition to the RUNNING state, resuming normal operation immediately following the defined stop location.</td>
</tr>
<tr>
<td>HOLDING</td>
<td>ACTIVE</td>
<td>ABNORMAL 11</td>
<td>The procedural element has received a HOLD command and is executing its HOLDING logic to put the procedural element or equipment entity into a known state. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the HELD state.</td>
</tr>
<tr>
<td>HELD</td>
<td>PAUSED</td>
<td>ABNORMAL 11</td>
<td>The procedural element has completed its HOLDING logic and has been brought to a known or planned state. This state is usually used for a long-term stop. The procedural element or equipment entity is waiting for a further command to proceed.</td>
</tr>
</tbody>
</table>
### Table 10. S88 Procedure States for TCL Sequences (Continued)

<table>
<thead>
<tr>
<th>S88 Procedure State</th>
<th>TCL Sequence State</th>
<th>TCL Abnormal Status</th>
<th>S88 State Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STARTING</td>
<td>ACTIVE</td>
<td>ABNORMAL 11</td>
<td>The procedural element has received a RESTART command while in the HELD state. It is executing its restart logic in order to return to the RUNNING state. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the RUNNING state.</td>
</tr>
<tr>
<td>STOPPING</td>
<td>ACTIVE</td>
<td>ABNORMAL 9</td>
<td>The procedural element has received a STOP command and is executing its STOPPING logic, which facilitates a controlled normal stop. If no sequencing is required, then the procedural element or equipment entity transitions immediately to the STOPPED state.</td>
</tr>
<tr>
<td>STOPPED</td>
<td>INACTIVE</td>
<td>ABNORMAL 9</td>
<td>The procedural element has completed its STOPPING logic. The procedural element or equipment entity is waiting for a RESET command to transition to IDLE.</td>
</tr>
<tr>
<td>ABORTING</td>
<td>ACTIVE</td>
<td>ABNORMAL 3</td>
<td>The procedural element has received an ABORT command and is executing its ABORT logic, which is the logic that facilitates a quicker, but not necessarily controlled, abnormal stop. If no sequencing is required, then the procedural element transitions immediately to the ABORTED state.</td>
</tr>
<tr>
<td>ABORTED</td>
<td>INACTIVE</td>
<td>ABNORMAL 3</td>
<td>The procedural element has completed its ABORTING logic. The procedural element is waiting for a RESET command to transition to IDLE.</td>
</tr>
</tbody>
</table>
Appendix D  Generic Tag Configuration

Configuring Generic Tags on MOD Environment Displays

The MOD300 Generic Tag Configuration aspect is used to define tag properties for 800xA system tags that are not coming from MOD 300 so they can be displayed on the MOD 300 Environment displays. The Environment Importer configures the environment displays with MOD 300 tags imported by the MOD 300 tag importer. Tags are associated with MOD_ENV_BLOCKs, Figure 47. Other system tags, from a softpoint or an AC 800M controller for example, must be configured to fit the data model required by the Environment Displays.

The following loop types are supported by the MOD 300 Environment displays.

- Digital Loop.
- Analog 1 (Continuous, Indication) Loop.
- Analog 2 (Control, A/M) Loop.

Figure 47. Typical Environment Blocks
• Analog 3 (PID) Loop.

Refer to 800xA for MOD 300 Operation, (3BUR002418*), for a detailed description of the Environment Displays. The set of available properties that can be used in the MOD300 Generic Tag Configuration aspect is determined by what loop type is selected.

General Configuration Procedure

1. To begin, configure an OPC object for the generic tag. See System 800xA Configuration (3BDS011222*), for information on integrating OPC Servers into 800xA.

2. Add a new Generic OPC object in the Control Structure underneath the Generic OPC Server Network object.

3. The Properties of the Generic OPC Object are shown in the Control Connection Aspect. For this example, MEASURE and DAT/DATA1 will be used.

4. Add a General Properties aspect when state strings are required. Add this aspect to the OPC object. Any properties that will be used during properties translation needs to be included to emulate the desired MOD 300 tag properties. For example, AM and TRK could be defined to build an expression for OUT_MODE.

5. Add a Properties Translation aspect, if required. Add an expression as needed, and set the data type. For additional information on the Property Translations aspect, see the System 800xA Operations Operator Workplace Configuration (3BSE030322*). Figure 48, shows an example Property Translations aspect for OUT_MODE for an Analog 2 loop type.

6. Add a MOD300 Generic Tag Configuration aspect to a MOD_ENV_BLOCK.
   a. Set the Loop Type.
   b. Configure the properties of the loop type properties by selecting an appropriate Value (see Figure 49).
Figure 48. Property Translations Aspect, Configuration Tab
Figure 49. Setting Property Value for Generic Tag Configuration

Digital Loop Properties Reference

The elements used in a digital loop, Figure 50, are described in Table 11.
Figure 50. Digital Loop for MOD300 Environment Generic Tag Configuration

Table 11. Digital Loop Type Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOOPDESC</td>
<td>MOD 300 Loop Parameter Name: Loop Descriptor.</td>
</tr>
<tr>
<td></td>
<td>Data Type: String.</td>
</tr>
<tr>
<td></td>
<td>Value: Description of loop.</td>
</tr>
<tr>
<td>DEV_STAT</td>
<td>MOD 300 Loop Parameter Name: Device State.</td>
</tr>
<tr>
<td></td>
<td>Data Type: signed short integer.</td>
</tr>
<tr>
<td></td>
<td>Value: State values defined on Device Descriptor template.</td>
</tr>
</tbody>
</table>
### Table 11. Digital Loop Type Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DEV_CMND</td>
<td>MOD 300 Loop Parameter Name: Device Command.</td>
</tr>
<tr>
<td></td>
<td>Data Type: signed short integer.</td>
</tr>
<tr>
<td></td>
<td>Value: state values defined on Device Descriptor template.</td>
</tr>
<tr>
<td>DEV_MODE</td>
<td>MOD 300 Loop Parameter Name: Device Mode.</td>
</tr>
<tr>
<td></td>
<td>Data Type: binary 1 integer.</td>
</tr>
<tr>
<td></td>
<td>Value: MANUAL = 0, AUTO = 1</td>
</tr>
<tr>
<td>NORMSTAT</td>
<td>MOD 300 Attribute Name: Normal State.</td>
</tr>
<tr>
<td></td>
<td>Data Type: binary 1 integer.</td>
</tr>
<tr>
<td></td>
<td>Value: Normal = 0, Abnormal = 1. If an abnormal state exists, the</td>
</tr>
<tr>
<td></td>
<td>NORMSTAT attribute is set to 1 and the tag is displayed in reverse video</td>
</tr>
<tr>
<td></td>
<td>on the operational displays.</td>
</tr>
<tr>
<td>STATE_STRINGS</td>
<td>State strings associated with DEV_STAT.</td>
</tr>
<tr>
<td>COMMAND_STRINGS</td>
<td>Command strings associated with DEV_CMND.</td>
</tr>
<tr>
<td>DIGITAL_InAlarm</td>
<td>800xA System generated property. Digital loop is in alarm.</td>
</tr>
<tr>
<td>DIGITAL_IsUnAked</td>
<td>800xA System generated property. Digital loop alarm is Unacknowledged.</td>
</tr>
<tr>
<td>DIGITAL_Priority</td>
<td>800xA System generated property. Digital loop alarm priority.</td>
</tr>
<tr>
<td>DIGITAL_AlarmString</td>
<td>800xA System generated property. Digital loop alarm text.</td>
</tr>
<tr>
<td>TREND_PROPERTY</td>
<td>This is the log of the DEV_STAT value.</td>
</tr>
</tbody>
</table>
Analog 1 (Continuous) Loop Properties Reference

The elements used in an Analog 1 loop for continuous or indicator loops, Figure 51, are described in Table 12.

Figure 51. Analog1 Loop for MOD300 Environment Generic Tag Configuration
Table 12. Analog 1 (Continuous) Loop Type Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MEASURE</td>
<td>MOD 300 Loop Parameter Name: Measured Variable. A loop can have only one measured value. Data Type: Real. Value:</td>
</tr>
<tr>
<td>ENGUNITS</td>
<td>MOD 300 Loop Parameter Name: Engineering Units label. Data Type: String. Value: Description of loop.</td>
</tr>
<tr>
<td>LOOPDESC</td>
<td>MOD 300 Loop Parameter Name: Loop Descriptor. Data Type: String. Value: Description of loop.</td>
</tr>
<tr>
<td>NORMSTAT</td>
<td>MOD 300 Attribute Name: Normal State. Data Type: binary 1 integer. Value: Normal = 0, Abnormal = 1. If an abnormal state exists, the NORMSTAT attribute is set to 1 and the tag is displayed in reverse video on the operational displays.</td>
</tr>
<tr>
<td>MEASURE_InAlarm</td>
<td>800xA System generated property. Measure property is in alarm.</td>
</tr>
<tr>
<td>MEASURE_IsUnAcked</td>
<td>800xA System generated property. Measure property alarm is Unacknowledged.</td>
</tr>
<tr>
<td>MEASURE_Priority</td>
<td>800xA System generated property. Measure property alarm priority.</td>
</tr>
<tr>
<td>MEASURE_AlarmString</td>
<td>800xA System generated property. Measure property alarm text.</td>
</tr>
<tr>
<td>TRENDPROPERTY</td>
<td>This is the log of the MEASURE value.</td>
</tr>
</tbody>
</table>
Analog 2 (Control) Loop Properties Reference

The elements used in a Analog 2 loop for control or A/M loops, Figure 52, are described in Table 13. Since Analog 2 properties include Analog 1 properties, only those properties unique to Analog 2 are described.

Figure 52. Analog2 Loop for MOD300 Environment Generic Tag Configuration
Table 13. Analog 2 (Control) Loop Type Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT</td>
<td>MOD 300 Loop Parameter Name: Output Variable. Data Type: Real. Value:</td>
</tr>
<tr>
<td>OUT_MODE</td>
<td>MOD 300 Loop Parameter Name: Output Variable. Data Type: Signed one byte. Value: 0 = MANUAL, 1 = AUTO, 2 = TRACK, (Track applies to PID, A/M only).</td>
</tr>
<tr>
<td>OUT_HILI</td>
<td>MOD 300 Loop Parameter Name: Output High Limit. Data Type: Real. Value:</td>
</tr>
<tr>
<td>OUT_LOLI</td>
<td>MOD 300 Loop Parameter Name: Output Low Limit. Data Type: Real. Value:</td>
</tr>
<tr>
<td>HILIMIT</td>
<td>MOD 300 Loop Parameter Name: High limit (Overrange counts). Data Type: Real. Value:</td>
</tr>
<tr>
<td>LOLIMIT</td>
<td>MOD 300 Loop Parameter Name: Low limit (Under Range counts). Data Type: Real. Value:</td>
</tr>
<tr>
<td>HI_CONV</td>
<td>MOD 300 Loop Parameter Name: High Conversion Limit. This is the Engineering Units upper bound. Data Type: Real. Value: Real number.</td>
</tr>
<tr>
<td>LO_CONV</td>
<td>MOD 300 Loop Parameter Name: Low Conversion Limit. This is the Engineering Units lower bound. Data Type: Real. Value: Real number.</td>
</tr>
</tbody>
</table>
### Table 13. Analog 2 (Control) Loop Type Properties (Continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>OUTPUT_InAlarm</td>
<td>800xA System generated property. Output property is in alarm.</td>
</tr>
<tr>
<td>OUTPUT_IsUnAcked</td>
<td>800xA System generated property. Output property alarm is Unacknowledged.</td>
</tr>
<tr>
<td>OUTPUT_Priority</td>
<td>800xA System generated property. Output property alarm priority.</td>
</tr>
<tr>
<td>OUTPUT_AlarmString</td>
<td>800xA System generated property. Output property alarm text.</td>
</tr>
</tbody>
</table>

### Analog 3 (PID) Loop Properties Reference

The elements used in a Analog 3 loop for PID loops, Figure 53, are described in Table 14. Since Analog 3 properties include Analog 1 and 2 properties, only those properties unique to Analog 3 are described.
Figure 53. Analog3 Loop for MOD300 Environment Generic Tag Configuration

Table 14. Analog 3 (PID) Loop Type Properties

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETPOINT</td>
<td>MOD 300 Loop Parameter Name: Setpoint value.</td>
</tr>
<tr>
<td></td>
<td>Data Type: Real.</td>
</tr>
<tr>
<td></td>
<td>Value:</td>
</tr>
<tr>
<td>SPT_MODE</td>
<td>MOD 300 Loop Parameter Name: Setpoint Mode.</td>
</tr>
<tr>
<td></td>
<td>Data Type: Signed one byte.</td>
</tr>
<tr>
<td></td>
<td>Value: 0 = LOCAL, 1 = REMOTE, 2 = TRACK, 4 = RAMP.</td>
</tr>
</tbody>
</table>
### Table 14. Analog 3 (PID) Loop Type Properties (Continued)

<table>
<thead>
<tr>
<th>Property</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETPOINT_InAlarm</td>
<td>800xA System generated property. Setpoint property is in alarm.</td>
</tr>
<tr>
<td>SETPOINT_IsUnAced</td>
<td>800xA System generated property. Setpoint property alarm is Unacknowledged.</td>
</tr>
<tr>
<td>SETPOINT_Priority</td>
<td>800xA System generated property. Setpoint property alarm priority.</td>
</tr>
<tr>
<td>SETPOINT_AlarmString</td>
<td>800xA System generated property. Setpoint property alarm text.</td>
</tr>
</tbody>
</table>
Appendix E  TCL Batch Commands and Configuration to Support PDL

Introduction

The OCS configuration that is needed to support PDL for Information Manager nodes is described in this section. Configuration is required in the Unit Master templates to route the messages to the Connectivity Server, and in the Connectivity Server to receive those messages.

TCL Batch Commands

The following TCL batch statements are supported:

- STARTBATCH, ENDBATCH.
- TRENDON, TRENDOFF.
- RECORD
- REPORT

For detailed information on proper configuration of the TCL batch statements, please refer to 3BUR001980R3501, the Taylor Control Language User Guide. The TCL Batch statement REPORT is not documented in this User Guide and thus its configuration is documented below.

If the system includes 800xA Batch Management, do not use the STARTBATCH or ENDBATCH commands because 800xA Batch Management creates the PDL
It is possible to use the RECORD statement and the data will be routed to the PDL created by 800xA Batch Management. The database configuration described here is required in order to have the TCL_Record routed to the PDL created by 800xA Batch Management.

**TCL Batch Command Configuration - REPORT**

The REPORT command accepts six parameters:

- ReportName
- Printer
- PrintOption
- SaveOption
- BatchID
- UnitID

All the above parameters require an entry to compile.

**REPORT (ReportName, Printer, PrintOption, SaveOption, BatchID, UnitID)**

Note that not all parameters are required to support 800xA for MOD applications. For use with 800xA for MOD 300, the *ReportName, BatchID, and UnitID* parameters must have correct input. The *Printer, PrintOption, and SaveOption* parameters must have **NULL** options for proper execution of the statement in 800xA for MOD 300 applications. These parameters are still required because they support older TCL applications.

An example REPORT command for an 800xA for MOD 300 application is as follows.

Configuration to Support PDL

Report Builder (REP_BLDR) Database Object

When configuring the MOD 300 database (using AdvaBuild Basic Functions), insert a Report Builder (REP_BLDR) database object as a child of the GENERICID object that represents the node where Reports will run (Figure 54). The Report Builder object attributes can be left at their default values. No further configuration is required for this object.

The Report Builder (REP_BLDR) object must reside in the Connectivity Server node where the OPC AE Server is configured.

Figure 54. REP_BLDR Database Object in Database Hierarchy

Report ID Fields in Unit Master Database Objects

TCL RECORD statements record data for TCL units. To ensure that production data recorded for a TCL unit are sent to the appropriate Reports package, associate the
TCL unit with the node where Reports Services runs. This is done by configuring the REPORT ID fields on the Unit Master objects.

Each Unit Master object that represents a unit where batches run must specify the name of the Report Builder object. The Unit Master object is a child of either a GENERICD or Controller object where the unit resides.

The REPORT ID is the object ID of the Report Builder database object for the node where Reports runs. This object ID must be entered in the REPORT ID field exactly as it appears in the AdvaBuild template view of the Report Builder database object. For redundant recording, specify a second Reports node in the REPORT IT 2 field.
Appendix F  Customized Data for Backup

Introduction

This appendix identifies specific settings and helps ensure that data is backed up to facilitate upgrades and/or node failure.

PAS Software Communication Settings

After installing and setting up PAS, it is necessary to record the communication settings of each node for future system maintenance needs. Refer to the System 800xA Post Installation (3BUA000156*) instruction for more detailed information.

For typical updates and upgrades where PAS is uninstalled and a new revision is installed you will be able to follow the instructions to revert to saved settings.

If a new node is installed or an upgrade to a new operating system occurs, the user will not have the saved communication settings to revert to which is why it is needed to enter the settings that were documented when the system was installed.

When PAS is reinstalled, the Reverse Time Sync setting is set to disabled, overriding the previous setting. The Reverse Time Sync setting should be recorded before uninstalling PAS so it can be manually restored.

To re-enable Reverse Time Sync:

1. Click **ABB Start Menu > ABB Industrial IT 800xA > PAS Management > PAS Control**.
2. Click on the **Advanced** button to open the advanced options.
3. Select **Enabled** from the drop down menu for Reverse Time Sync.
4. Click **OK**.
Operational Registry Settings

Table 15 describes operational registry settings that can be customized by the user. Record all the Key and Value settings if changed from their default setting and record that data for each node for future use.

Operational registry settings will need to be reentered if a new node or operating system is installed. They will not need to be reentered if updating or upgrading software on the same node or if using an image copy to replace a node.

Table 15. Operational Registry Settings

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>HKEY_LOCAL_MACHINE &gt; SOFTWARE &gt; Wow6432Node &gt; ABB &gt; AdvOPCAEServer &gt; config</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Automatic TCL Reply Delay Enabled</td>
<td>1</td>
<td>A value of one indicates that the AE server automatically determines the TCL Reply Delay.</td>
</tr>
<tr>
<td>Default TCL Reply Delay (ms)</td>
<td>2000</td>
<td>When the Automatic TCL Reply Delay logic is enabled, this value is used for the delay when a delay is required on a node.</td>
</tr>
<tr>
<td>LogFileSize (kB)</td>
<td>Default value of 1024.</td>
<td>The minimum size is 1024. Cannot be set smaller than that. May be set larger for debug support use.</td>
</tr>
<tr>
<td>Redundant Partner</td>
<td>Blank if no Redundant Connectivity Servers is specified during installation. Otherwise this is the name of the redundant connectivity server.</td>
<td>Record value and restore if changed from default.</td>
</tr>
<tr>
<td>Standalone Server</td>
<td>0 = No. 1 = Yes, this node is a standalone server.</td>
<td>Record value and restore if changed from default. The StandAlone option is only used for demo purposes.</td>
</tr>
</tbody>
</table>
### Table 15. Operational Registry Settings (Continued)

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
</table>
| **UseGlobalAck** | 0 = Off (Default, No Global Acknowledge used)  
1 = On (Global Acknowledge is used) | Record value and restore if changed from default |
| **Fixed Rate Threshold (ms)** | Default is 8000 | Subscriptions requested slower than 8000 milliseconds make it through to OMF as less than 8 second subscriptions. |
| **Use Composite Update Time Change** | 0 = Disabled. Composite attributes not used unless specified directly using 
* _COMP.  
1 = Enabled. Use * _COMP when subscription rate is greater or equal to Composite Update Time Change (ms) value.  
Default is 0 Disabled (Off). | Enables or disables the use of the Composite Update Time Change registry entry. |
| **Fixed Rate OPC DA Server (ms)** | > =2000 ms  
Default value is 2000 ms. | Record value and restore if changed from default. Do not set to a value less than 2000. |
| **Use Fixed Rate Server** | 0 = Fixed Rate Server not used.  
1 = enabled. Fixed Rate Server used.  
Default value is 1 | Record value and restore if changed from default. Do not set to ‘use standard subscription’ unless directed by ABB Support Engineering |
### Table 15. Operational Registry Settings (Continued)

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HKEY_LOCAL_MACHINE &gt; SOFTWARE &gt; Wow6432Node &gt; ABB &gt; AdvOPCDAServer &gt; config (continued)</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Composite Update Time Change (ms)</td>
<td>Threshold at which *COMP is used. Works in conjunction with the Use Composite Update Time Change key. This should be set to a value greater than the “Fixed Rate for OPC DA Server”. E.g. Fix Rate for OPC DA Server set to 2000ms, value for Composite Update Time Change should be 2001ms or greater. <strong>Note:</strong> History Logs should be configured such that the sample interval is &gt;= Composite Update Time Change. Default is 4000 ms.</td>
<td>Set to a value greater than the 'Fixed Rate OPC DA server (ms)'. Restore value if default changed</td>
</tr>
<tr>
<td>Device Descriptor Set Refresh</td>
<td>Normally 0, the default 1, forces refresh. Set to 1 after descriptor set is modified in AdvaBuild to update cache. Then reset back to 0</td>
<td>Will be restored to default.</td>
</tr>
<tr>
<td>LogFileSize (kB)</td>
<td>Default value of 1024.</td>
<td>The minimum size is 1024. Cannot be set smaller than that. May be set larger for debug support use.</td>
</tr>
<tr>
<td>Standalone Server</td>
<td>0 = No. 1 = Yes, this node is a standalone server.</td>
<td>Record value and restore if changed from default. The StandAlone option is only used for demo purposes</td>
</tr>
<tr>
<td>Special Algorithm Descriptions Refresh</td>
<td>Normally 0, the default 1, forces refresh. Set to 1 after the algorithm descriptor set is modified in AdvaBuild to update cache. Then reset back to 0</td>
<td>Will be restored to default</td>
</tr>
</tbody>
</table>
### Table 15. Operational Registry Settings (Continued)

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HKEY_LOCAL_MACHINE &gt; SOFTWARE &gt; Wow6432Node &gt; ABB &gt; MOD Add-on &gt; config</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ChangeableColorArea GroupDisplay</td>
<td>REG_DWORD parameters used for setting colors</td>
<td>Record these values if you change from the defaults. Refer <strong>MOD 300 Display Colors</strong> on page 107 for more information.</td>
</tr>
<tr>
<td>LimitColorAreaGroup Display</td>
<td>REG_DWORD parameters used for setting colors</td>
<td></td>
</tr>
<tr>
<td>StaticColorAreaGroup Display</td>
<td>REG_DWORD parameters used for setting colors</td>
<td></td>
</tr>
<tr>
<td>ReadOnlyAreaGroup Display</td>
<td>0=no. Default 1=yes. Displays are read only</td>
<td>Record value and restore if changed from default</td>
</tr>
<tr>
<td>AreaGroupDisplay OPCUpdateRate (ms)</td>
<td>Default value is 2000</td>
<td>Do not change.</td>
</tr>
<tr>
<td><strong>HKEY_LOCAL_MACHINE &gt; SOFTWARE &gt; Wow6432Node &gt; ABB &gt; System Services &gt; DxBase &gt; dxTimeSync</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reversed_Synch_Mode</td>
<td>0 = disable. Default (RTA sets the PC time) 1 = enable</td>
<td>Value is set to 0 whenever PAS is installed. Record and set value as required for Connectivity Server node.</td>
</tr>
<tr>
<td><strong>HKEY_LOCAL_MACHINE &gt; SYSTEM &gt; CurrentControlSet&gt; Services &gt; W32Time &gt; Parameters</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NtpServer</td>
<td>time.windows.com,0x1</td>
<td>To be set on the Connectivity Server node when reverse time synch is disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To be set on the Domain Controller node when reverse time synch is disabled</td>
</tr>
<tr>
<td></td>
<td></td>
<td>To be set on the 800xA for MOD 300 client node when reverse time synch is disabled</td>
</tr>
</tbody>
</table>
Fixed Registry Settings

Table 16 describes fixed registry settings that should not be customized by the user. These registry settings are set by 800xA for MOD 300 and should not be changed unless instructed by ABB Technical Support.

Table 16. Fixed Registry Settings

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HKEY_LOCAL_MACHINE &gt; SOFTWARE &gt; Wow6432Node &gt; ABB &gt; AdvOPCAEServer &gt; config</strong></td>
<td></td>
</tr>
<tr>
<td>CCF Require Ack for Clear Messages</td>
<td>0 = Off (Default, no acknowledgment of cleared message) 1 = On (CCF requires a cleared message to be acknowledged)</td>
</tr>
<tr>
<td>Diagnostic Engineer Messages as Condition</td>
<td>Default is 0.</td>
</tr>
<tr>
<td>LogFileEnabled</td>
<td>Default value of 1.</td>
</tr>
<tr>
<td>LogFileName</td>
<td>Default value of AdvOPCAEServerTrace.log</td>
</tr>
<tr>
<td>LogFilePath</td>
<td>Default value of C:\Temp</td>
</tr>
<tr>
<td>RTAB Watchdog (secs)</td>
<td>Default is 60 seconds.</td>
</tr>
<tr>
<td>Send Global Ack to LCP</td>
<td>0 = Off (Default, Global Acknowledge is not sent) 1 = On (Global Acknowledge is sent to LCP)</td>
</tr>
<tr>
<td>Send Global Ack to MOD 300</td>
<td>0 = Off (Default, Global Acknowledge is not sent) 1 = On (Global Acknowledge is sent to MOD 300)</td>
</tr>
<tr>
<td>Store Ack for Condition Events</td>
<td>Default is 1</td>
</tr>
<tr>
<td>TCL Reply Delay (ms)</td>
<td>The TCL Reply Delay that will be used by this node’s MOD 300 AE server to prevent from sending two TCL replies to a controller simultaneously. This setting has been superceded by the Automatic TCL Reply Delay Enabled registry entry. Should not be changed without ABB support.</td>
</tr>
</tbody>
</table>
Customizing Objects

If objects are customized within the system, record these changes. This can be done by documentation, exporting changes with the Import/Export Tool, and System Backup. Following an update or upgrade, those changes must be re-implemented on the objects that may be changed by the upgrade.

Custom Color Definitions

When upgrading software, the color definitions will be changed to the default Common Alarm Color Definition settings. If the previous configuration referenced
something other than these Common Colors, then that change will need to be made. The MOD 300 Alarm Color Definition is not selected by default when performing an upgrade.

Changes made to the MOD Alarm Lists can be modified in the Library Structure, specifically in the MOD Alarm and Event Configurations Group.
Revision History

Introduction

This section provides information on the revision history of this User Manual.

The revision index of this User Manual is not related to the 800xA 6.0 System Revision.

Revision History

The following table lists the revision history of this User Manual:

<table>
<thead>
<tr>
<th>Revision Index</th>
<th>Description</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>First version published for System 800xA 6.0</td>
<td>August 2014</td>
</tr>
<tr>
<td>A</td>
<td>Updated for System 800xA 6.0.1</td>
<td>October 2015</td>
</tr>
<tr>
<td>B</td>
<td>Updated for 800xA 6.0.3</td>
<td>September 2016</td>
</tr>
</tbody>
</table>

Updates in Revision Index A

The following table shows the updates made in this User Manual for System 800xA 6.0.1:

<table>
<thead>
<tr>
<th>Updated Section/Sub-section</th>
<th>Description of Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>Appendix F Customized Data for Backup</td>
<td><em>Table 15 Operational Registry Settings and Table 16 Fixed Registry Settings have been updated with minor changes.</em></td>
</tr>
</tbody>
</table>
The following table shows the updates made in this User Manual for 800xA 6.0.3:

<table>
<thead>
<tr>
<th>Updated Section/Sub-section</th>
<th>Description of Update</th>
</tr>
</thead>
</table>
| Section 2 Administration    | Removed the following subsections:  
- Create System  
- Add System Extensions  
For Add Server and Client Nodes subsection:  
Replaced the entire section with a reference to *800xA Installation, Update and Upgrade Getting Started* manual. |
A
AC410 object 72
AC460 object 72
AC460MOD object 73
Access Rights 31
Acknowledge, Global 161
AdvaBuild 13
AdvaBuild Structure Builder 18
AdvaBuild Template Builder 18
Advant OCS 13
ADVANT_D2D object 72
ADVANT_STATION object 72
AdvMODBatchOPCDAServer.MODBatchOPCD
AServer.1 22
AdvOPCDAServer.OPCDAServer.1 22
AINPUT FCM 73
Alarm Colors 106
Alarm Routing 66
ALIASID 132
AliasID 134
Ancestors, Show 39
AOUTPUT FCM 73
AREA object 72
Array Element Reference Format 126
Aspect
  Audit Trail Config 70
  Graphic Element 90
  MOD300 Alarm Colors 106
  NLS Resource Manager 108
  OPC Data Source Definition 65
  Security Definition 76
  Service Group Definition 26
  Trend Template 104
Aspect Directory 29
Aspect Server 17
Audit Event classes 70
Audit Trail Config 70
Auto Start, GENERICD Object 115
Browse Form 39
BUC object 73

B

C
CCF 13
CCF Alarm Logger 69
CCF object 73
CCF Require Ack for Clear Messages 164
Client Application 17
CNTRLLER object 73
Colors, Trend 104
Colors, Alarm 106
Composite 63
Composite (_COMP) Configuration for MOD
  Trend Data Quality and History Data 63
Connectivity Server 17, 111
CONSOLE object 72
CONT_SS object 72
CTRL_BLOCK object 73

D
D2F object 73
Database
  MOD 300 18, 111
  References, Syntax 123
DBVAR (data base variables) 132
DCN 13
DCN Address 114
DCN_DCN object 72
DEV_LOOP object 73
Index

Device Descriptor Set Refresh 162
Diag Logger 69
DO Rate, GENERICD Object 115

E
ENVDUMP Utility 46
Environment Importer 45
Environment, Default, GENERICD Object 115
Event Logger 69

F
FCM 73
FCM Parameter Reference Format 125
Filled Plot, TCL Array 96
Find 37
Find Display 37
Find Now 38
Fixed 63
Fixed Rate MOD 300 OPC Server
  Configuration 63

G
Generic Tag Configuration 141
GENERICD Object 18, 111, 113
GENERICD object 72
Global Acknowledge 161
Graphic Element 90

H
Hybrid Line Array 96, 102
Hybrid Plot 96

I
Illegal Character 31, 82
Import 35, 39
  Requirements 17
  Tag Objects 24
  Tags 29
Import Display 35, 51

L
LL_DEV object 73
Log Configuration Aspect 105
Log File 40 to 41, 53
Logger Configuration 69
Logger, Message Center 118
Loop Parameter Reference Format 124
Loop Type, Changes 119
LOOP_DEF object 73

M
Master Timekeeper Node 18
Memory
  MTOS Memory Pools 119
Message Center 116
  entries for Alarm/Event Loggers 118
MOD 300 Database 18, 111
MOD 300 Plot Aspect 95
MOD 300 Utility 67
MOD OPC Server Network 20
MOD OPC Statistics Aspect 68
MOD Templates 71
MOD_DB object 71 to 72
MOD_DValue Graphic 90
MOD_ENV_AREA object 83
MOD_ENV_BLOCK object 85
MOD_ENV_GROUP object 84
MOD_PHASE 131
MOD_SEQUENCE 74, 76
MSG_ROUT object 118
MSQR FCM 73
MTOS Memory Pools 119

N
Navigator Display 33 to 34, 48, 57
NLS Resource Manager 108
Node Type, GENERICD Object 115

O
Object Status 42, 50, 58
Object Types 30, 73
OMF Object ID’s 119
OPC DA Service Group 31
OPC Data Source 65
OPC Data Source Definition 65
OPC Group 30
OPC Server 62
OPC Server ProgID 22
OPCAE Service 26
OPCDA Service 25 to 26
Open, Environment Text File 46
Operational registry settings 160
Options 41
Options Display 43, 50, 58

Parameter Change 70
PAS 159
Password 32, 56
Permissions for MOD Objects 76
Phase Importer 55
Physical Device, GENERICD Object 114
PID FCM 73
Plot, TCL Array 96
Project Name 32, 56
Properties Translation 142

Recipe Item Reference Format 127
Reference Formats
  Array Element 126
  FCM (Tag-FCM Access) 125
  Loop Parameter 124
  Recipe Item 127
  TLL Parameter 128
Registry settings 160
Report, Tags Totals 67
Reverse Time Synchronization (RTS) 18
Routing, Message Destination 118
RTAB Watchdog(secs) 164 to 165

RTS 18

S
S88Phase.tcl 131
Search 38
Security Definition 76
Security System 19
Server Name 32, 56
Service Group Definition 26
Service Groups 25
Service Provider Definition 27
Service Structure 26
Settings 159
  Registry 160
Show Ancestors 39
Single Object 35
Software Name, GENERICD Object 115
Special Object Names 75
Start System 61
Statistics Aspect 68
Status Bar 33, 57
Structure Builder 18
Subelement, OCS Graphic
  MOD_DValue 90
System Structure 19

T
Tag Objects 24
Tag Objects, Totals Report 67
Tag Status 51
TCL 13
TCL Array Plot 95
TCL Message Logger 69
TCL Real Number Array 102
TCL Reply Delay(ms) 164
Template Builder 18, 113
Templates, MOD 71
Templates, MOD Configuration 73
Time, Service Object 27
TLL 13
Index

TLL Elements 42
TLL Parameter Reference Format 128
ToolBar 33
Tree View Tool Tip 42, 49
Trend Buffer 105
Trend Colors 104

U
UNITMAST (MOD_UNIT) object 73
UNITVAR 132
User Acknowledge 69
User LogOn/Off 69
User Name 32, 56
User Type 19
Utility, MOD300 67

V
VAR. (local variables) 132