L13 – Studio 5000 and Logix Advanced
Advanced Logix Topics with a v31 Preview
Lab Overview

Explore the Studio 5000® Design Environment

- Learn about new usability enhancements that are available in within Studio 5000® Logix Designer.
- Learn how to improve efficiencies using modular programming
- Learn about new features that enhance modular programming within Studio 5000® Logix Designer
- Hands-On experience with a pre-release* V31 of Studio 5000® Logix Designer

*Note: Pre-Release information is subject to change.
Studio 5000 Logix Designer®

What’s New?

NEW RELEASE

- Common look and feel across all Studio 5000® applications
- Improved workflows for multi-monitors, tabbed views, quick navigation
- Modernization of programming language editors
- Clearly convey state and identification of errors
Increased usability with tabbed windows

- All editing windows are now tabbed
  - Logic, Parameters, Trends, AOPs, UDTs
- Quick close capability
- Reduced number of clicks
- Re-ordering of tabs

Organize your view with “tab groups”

- Users can create custom “tab groups”
- Allows for organization key routines, tags, trends, etc.
Multi-Monitor Support

Organize your view the way that you like in a productive manner
Multi-Monitor Support

Organize your view the way that you like in a productive manner
Multi-Monitor Support

Organize your view the way that you like in a productive manner
Overview
A modernized structured text editor packed with features for productive programming and editing

- Line numbers and bookmarks
- Descriptive tool tips and syntax highlighting
- Multi-line select and mouse scrolling
- Change and verify bars
- Collapsible code segments
- Inline value monitoring
- Code snippets and smart indent capabilities

Benefits
- Modern programming capabilities
- Increased productivity
- Efficient code development and editing
Logix Designer®: Structured Text Editor
Enhancement: Change Bars

106 | yyTest := l;
107 | If yyMR2 Then
108 |  yyI := 0;
109 |  yySample_WK2 := 0;
110 |  yyTest_Spd_Ref := xxTest_Spd_SET / xxConstant_RPMPerFPM * xxBuildUpRatio2;
111 |  End_If;

106 | yyTest := l;
107 | If yyMR2 Then
108 |  yyI := 1;
109 |  yySample_WK2 := 1;
110 |  yyTest_Spd_Ref := xxTest_Spd_SET / xxConstant_RPMPerFPM * xxBuildUpRatio2;
111 |  End_If;

// Mechanical WQ data sampling //
115 | If xxTest Then
116 |  yyTest_Spd := xxTest_Spd_SET / xxConstant_RPMPerFPM * xxBuildUpRatio2;

// Calculate WQ for the upper guide roll //
120 | If xxTest and (yyTest_Spd >= xxMin_Spd_SET) then
121 |  For yy := 0 to 14 do
122 |     yySample_WK2 := yySample_WK2 + yySample_WK2[yyI];
123 |  End_for;
124 |  yySample_WK2 := yySample_WK2 / 15;
125 | If yySample_WK2 = 0 then
126 |     yySample_WK2 := 0;
Editor Improvements: Function Block

Overview
Improvements to Function Block editor for increased usability
- Default sheet size changed to 11x17 (landscape) (V30)
- Ability to change sheet size while online
- Forcing I/O tags from Context Menu
- Direct operand value modification

Benefits
- Level setting editor experiences across all editors for consistent and efficient workflows
Logix Designer®: Function Block Editor
Default Sheet Size and Layout Changes

Before the default size was
8.5x11 Portrait

Now the default size is
11x17 Landscape
Modification of the FBD sheet size while online
Modification of the FBD Instruction

**STEP 1**
Right Click, Select Monitor “input1”

**STEP 2**
Tag Monitor opens, select value, change value

**STEP 3**
Return to FBD Diagram

Before
Modification of the FBD Instruction

STEP 1
Click the Number

STEP 2
Change Value, Press Enter

After
Logix Designer®: Ladder Editor
Instruction Options for Vertical Space
Logix Designer®: Ladder Editor
Instruction Options for Vertical Space

[Diagram of Ladder Editor with settings for instruction options for vertical space]
Logix Designer®: Code Editors
Standard Indication of Errors Makes Troubleshooting Easy

Clear indication of where the error exists in the code editors
Logix Designer®: Output Window
New Message Filtering Capabilities

Filter capability added to the error window for quick identification.

Errors

1 Error(s) | 1 Warning(s) | 3 Messages

Verifying program: Utilities...
Verifying routine 'Lonely_Routine' of program 'Utilities'
Warning: Lonely_Routine: Routine cannot be reached by the main routine: 'Lonely_Routine of Program Utilities'
Verifying routine 'Main' of program 'Utilities'
Error: Rung 0, CONCAT, Operand 1: Missing operand or argument.
Complete - 1 error(s), 0 warning(s)

Errors

1 Error(s) | 1 Warning(s) | 3 Messages

Warning: Lonely_Routine: Routine cannot be reached by the main routine: 'Lonely_Routine of Program Utilities'
Complete - 1 error(s), 1 warning(s)
In a lab we will review the following Usability Enhancements from v28 and greater:

- Delete a Program without having to Unschedule
- Automatically Insert a Header in a Structured Text Routine
- Detect a Lonely Routine with Verify
- New On-Line Power Rail Display
- Noted v31 Enhancements
Logix Designer Compare Tool

Tool updated for the new features of Studio 5000®

- Find changes between two different Studio 5000® Logix Designer applications
  - Creates a report of the application differences
  - Can compare Export Files! (Rung, AOI, Routine, Program, etc..)
- Merge changes between Studio 5000® Logix Designer applications
New Ways to Access the Module Object
What’s the Module Object?

- The Module Object provides status information relevant to a module

- Attributes –
  - Entry Status
  - Fault Code
  - Fault Info
  - FW Supervisor Status
  - Force Status
  - INSTANCE
  - Status Indicator
  - Mode

- Path

FYI That’s New
New Ways to Access the Module Object

New Attribute to the Module Object: Path

- Use GSV to retrieve the Module Communication Path
  - SINT Array

- Use an SSV to Set the Msg Path

- Possible Re-use of Message for Multiple modules in an Add-On Instruction
  - Coding required
Access to Module Object
Inside an Add-On Instruction

- Full Access to the Module Object inside the Add-On Instruction
  - "Module" Data type as InOut Parameter (Add-On Instruction & Programs)
  - Allows you to point to the module from within an Add-On Instruction Definition
Partial Import Online (PIO) Enhancements

- Easily copy content across projects or within project
- Manage references (Tag Aliases, Program Connections)
- Manage dependencies (Add-On Instruction’s UDT’s and Tags)
- Manage conflicts (Overwrite or Use Existing)
- Optionally include/exclude data values
- Supports online and offline workflows
Partial Import Online (PIO) Enhancements

- Improved Search and Replace
  - Anything you can change via PIO dialogs
  - Wildcards, Replace All

<table>
<thead>
<tr>
<th>Find What</th>
<th>Replace With</th>
<th>Text Found</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tk1_*_001</td>
<td>PRE_*_010</td>
<td>Tk1_me_001</td>
<td>PRE_me_010</td>
</tr>
<tr>
<td>Tk3_*</td>
<td>Tk4_*</td>
<td>Tk3_unit</td>
<td>Tk4_unit</td>
</tr>
<tr>
<td>*001</td>
<td>*002</td>
<td>Mi_2001</td>
<td>Mi_2002</td>
</tr>
</tbody>
</table>

Find / Replace

- Find What: Tk1_*_001
- Replace With: PRE_*_010
- Use Wildcards
- Search current view only
- Direction: Up
- Find Within: Import Name, Final Name, Description, Alias For, Data Type, Parameter
Program Parameters
Program to Program Communications – Before

Previously

Controller Tags Act as “Gateway”

Data

Prog_01

Prog_02
Program Parameters

Program to Program Communications – After

 Introduced v24

Controller Tags

Direct Communication Between Program Folders

Data

Prog_01

Prog_02

Program Parameters

Program to Program Communications – After

Introduced v24

Controller Tags

Direct Communication Between Program Folders

Data

Prog_01

Prog_02
Program Parameters
The Composition of a Program (Before and After)
Why Add a Parameter Interface?

- Better understanding of interaction between programs
- Allow direct communication between programs
- Introduce a larger modular object in Logix Designer
Logical Organizer

Programs Span the Tasks

Logically Organized
Advanced Topics Labs

- **Lab 1 – Usability Enhancements**
  - Preview of new features that will be available in v31.
  - Review of new usability enhancements that are included in v28 and greater.

- **Lab 2 - Logical Organizer**
  - The Logical Organizer lets you organize your Logix application any way you would like without affecting the application execution.

- **Lab 3 - Compare and Merge Tool**
  - Updated for compatibility with the new features of Studio 5000®

- **Lab 4 - Add-On Instruction Access to Module Object**
  - New Logix feature to allow one Add-On Instruction to be accessing the module object
  - New GSV Module Object Path Attribute

- **Lab 5 - Partial Import Online**
  - A utility that allows the user to view/handle collisions, rename items and configure connections
Other Advanced Topics Labs

- **Lab 6 - Program Parameters**
  - Code modules at the Program level linked together by clearly defined input and output parameters

- **Lab 7 - Add-On Instructions (Add-On Instruction)**
  - Add-On Instruction Creation
  - Reusing an Add-On Instruction
  - Add-On Instruction signatures

- **Lab 8 - Using an SD Card with a Logix Controller**
  - Hands on example of reading and writing data to the SD Card from a Logix application
  - Have one of the instructors give you an SD Card before starting the lab.

- **Lab 9 - Logix Controller Change Log**
  - Examples of what is logged
  - How to read the log file on the SD card
Pick and Choose Your Lab Sections

Lab allows the user to run only the sections they are interested in. Lab Sections can be run in any order.

- Users can run any lab sections they would like
- Users can skip lab sections they are not interested in
- Estimated Lab completion times are located in the Table of Contents
  - Using these estimated lab completion times users can gauge how many of the labs they can complete for the allotted time available

There is not enough time in this session to complete all of the labs
Thank You!