Mexml

Machine Edition program analysis and reporting through the inspection of XML project files

Joalah Designs LLC

The Problem

- GE IP's Machine Edition enables you to create complex PLC programs with 1000's of variables
- But it has few quality assurance or code metric tools
- The only "real" tool is the Database Report which only provides "canned" reports that can't be exported from ME
- Some things these reports **can't** do include listing:
  - Declared I/O variables that are not used
  - Program blocks that are not called
  - Variables that are read but not set
  - Variables that have overlapping addresses

In Perspective

The following block and variable counts were observed in a series of real-world PLC projects:

Program	Block Count	Variable Count
Program “A”	85	9,800
Program “B”	106	9,400
Program “C”	140	16,300
Program “D”	116	15,000
Program “E”	112	12,700

With program sizes like these, it is impossible to manually keep track of every block or variable’s usage.

The Solution: Mexml

- Mexml reads XML files exported from Machine Edition and builds up a view of variable and program block usage. Two types of reports can be generated from this information
- Program Structure reports describe the overall structure of various aspects of the program itself
- Variable Usage reports record if/how a particular varibable was used within the program
- The reports can be generated in either plain text or CSV formats, allowing for easily incorporating results into documentation or using them for further processing

Program Structure Reports

Reports on the structure of the program and includes items such as:

- Program block usage (used/not used)
- Program block dependency and call order
- Variables that are mapped to the same address
- Variables with overlapping addresses
- Variables in Function Blocks that are global to the FB
- Program instructions that access addresses outside of a variable’s size
- Chains of variables aliased to parents
- Text representation of Ladder Logic and Structured Text instructions

Variable Usage Reports

Reports if/how a variable was used and can be generated using any combination of:

Access	– Read, Write, Input Scan, Output Scan etc.
Memory	– %I, %Q, %M, %R, Symbolic etc.
Type	– Local, Global, Input, Output etc.
Name	– As described by a regular expression
Address	– As described by a regular expression
Description	– As described by a regular expression

For example, combining Access and Memory criteria, or Type, Name and Address criteria.

Variable Usage Report Example:

Reports where and how a variable has been accessed:

```
                                  Access
Name                Type Address 1234567 Description
------------------- ---- ------- ------- ----------------
...
Number1             DINT %R00021 IR----- This is number 1
Number2             DINT %R00122 -RW---- This is number 2
Data_5              INT  %R00154 -R----- Data number 5
Data_6              INT  %R00755 -RW---Q Data number 6
...
```

Where the Access columns are:

1. Is part of an Input Scan (I)
2. Has been explicitly Read (R)
3. Has been explicity Written (W)
4. Was an array element that was read with a calculated index (S)
5. Was an array element that was written with a caclulated index (X)
6. Was accessed as a part of a Function Block (B)
7. Is part of an Output Scan (Q)

Program Structure Report Example:
Mapped Addreses

List the variables mapped to the same addresses:

```
############################
  Mapped Address variables
############################
  %M00105 WORD InputAsWord
    %M00105
    %M00106 BOOL LostKeyBit
    %M00107
    %M00108 BOOL OnFireBit
    %M00109 BOOL SinkHoleBit
    %M00110
    %M00111
    %M00112 BOOL MorningBit
    %M00113
    %M00114
    %M00115
    %M00116
    %M00117
    %M00118
    %M00119
    %M00120
```

Program Structure Report Example:
I/O Listings

List the hardware I/O to variable mappings:

```
Slot  Card
----  ----

 ...
 4    IC693MDL241
  Main %I00001 16 BOOL Reference Address
       %I00001    BOOL RSplines        Reticulate the splines
       %I00002    BOOL Starvation      The cafeteria isn't open
       %I00003    BOOL LostKey         Can't find the operating key
       %I00004    BOOL SuperPowers     Super powers have been granted
       %I00005    BOOL OnFire          Equipment is on fire
       %I00006    BOOL SinkHole        Sink hole has opened up beneath room
       %I00007    BOOL LotteryWin      Operator has won the lottery
       %I00008    BOOL I00008
       %I00009    BOOL PumpA_Running
       %I00010    BOOL PumpB_Running
       %I00011    BOOL ControlRmEmpty  No operators in the control room
      ...
```

Some benefits/uses of Mexml

- Ensure that all variables are initialized
- Ensure that all I/O points have been utilized
- Ensure that data tranfers don’t spill over their bounds
- List the variables that make up the external interface to a set of program blocks
- List the program block dependency and call order
- List all the Boolean variables mapped onto Word data types
- List the variables in every EGD

Requirements

- Mexml requires the .Net 4.0 client library, and runs on either 32 or 64 bit Windows systems, (but runs as a 32 bit process on 64 bit systems)
- Machine Edition Versions 7.0, 8.0 or 8.5

NOTE: Not all features work with 90-30 projects that have been imported into Machine Edition. Most notable is that these projects retain an incompatible EGD and Hardware file format.

Licensing

- Mexml requires a license before it can be used
- Trial or Standard licenses are available
- A free Trial license is supplied with Mexml, but limits the programs functionality
- Licenses do not expire and can easily be transferred from one computer to another
- A significant update in the Mexml version will require a license renewel *only* in order to use the new version

The bottom line

Mexml gives you the power to better track and report on many important aspects of your PLC projects, that until now have been hidden from you in a sea of raw data.

Whether it’s by helping you document existing systems, or ensuring that you have accounted for all the resources in a new program, Mexml enhances your productivity and helps you better validate your projects.

Better productivity and validation means that projects are completed sooner, with fewer mistakes and of course reduced costs.

Contact Us

For more information regarding Mexml please eMail Joalah Designs at:

Info@JoalahDesigns.com

Suggestions and requests are also welcome!