Tips and Tricks

In This Appendix...

Tips and Tricks

C-more tag import

To easily get your tags from a Do-more! Designer Project into a C-more HMI, in Do-more! Designer go to File –> Export -> Element Documentation. Pick C-more Do-more! Driver Format WITH Structure Fields.

The most common issue with importing tags into a C-more HMI is that you have not given all of your memory elements Nicknames. Without Nicknames C-more cannot make a proper tag out of your memory fields.

Pointers, Indexing and Arrays

In order to utilize a memory element as an array, it must be a Memory Block. Heap Items are singeltons and cannot be utilized as an array. To see what elements can be used as an array or to create a custom item, go to System Configuration -> Memory Configuration -> Memory blocks tab.

Addressing a Memory Block as an array is simple. Place brackets [ ] after the Memory Block name and use a V-memory element such as V0 as the index. IE D[V0], N[V5], UDT[V0], etc.

A handy trick for knowing how large the array is without hardcoding values is to use the DATAINFO instruction.

Bit of Word, Data type conversion and Casting

Getting bits out of a word in Do-more! Designer is done with a method called Casting. More information on Casting can be found in Do-more! Designer Help Topic DMD0309.

The basics of Casting a memory element are to use a ":" and an identifier or for Bit of Word, a number. IE D0:1 is bit 1 of memory element D0. All Bits start at zero.

To simplify Casting, you can use the Cast Builder. When on a field that supports a memory element, press the F9 key. Type your Source Element in the appropriate field and then in the bottom right corner, press the "Show Cast Builder" button. Click the appropriate radio buttons and then press "Select". It is that easy to cast a memory element.

Other methods for doing memory type conversion is by using the ladder commands PUBLISH and SUBSCRIBE. These will allow you to convert large areas of memory at once which is useful when you are talking to a Modbus or DirectLogic device that utilizes untyped memory.

Formatting Date and Time on an HMI with String Scripting

If you want your date and time to look nice on your HMI, put the data into a string using STRPRINT and the string scripting language. See Do-more! Designer Help Topic DMD0168 for a list of all string scripting commands.

MATH, Logical Operations & statistical functions

The Do-more! MATH instruction is a very powerful feature. Do-more! Designer Help Topic DMD0085 covers the lists of functions that are available.

In addition to doing Arithmetic and Trigonometry, it can also do Logical and Bitwise operations. A few minutes perusing this help topic can save you a lot of time.

The most common issue when doing Math with a Do-more! processor is knowing that you must promote the equation to Real (Floating Point) if you are using integers with a Real result. If the equation is integers, but you wish to have a result that is a Real number you must use TOREAL() or add a decimal point to one of the numbers; IE 2.0. See Help topic DMD0085 for more information on promoting numbers to Real format.

Simulation and PID Simulation

Do-more! Designer software comes with a separate Do-more! simulator application that runs the same Do-more! control engine that is in the Do-more! CPUs. You can connect/disconnect and upload/download to the simulator as you would with a Do-more! CPU. There is even a Run/Term/Stop mode switch and LED indicators for power, Run and communication activity. This is a great tool to test your code before you ever load it into your system.

The Simulator also has the ability to do simple PID simulation to learn how PID works!

See Do-more! Designer Help Topic DMD0234 for more information.

Symbolic Constants

Do-more! Designer supports the use of Symbolic Constants. A Symbolic Constant is where you assign a nickname to a hard coded value. IE Sunday = 0, Monday = 1, Tuesday = 3, etc. Do-more! Designer Help topic DMD0276 has more information on symbolic constants.

To assign a Symbolic Constant, go to Tools -> Documentation Editor and then press the "Symb" in the toolbar of the Documentation Editor tab. This can help make your program much easier to read and navigate.

You can add a nickname to a memory location that has a cast. For example you have a double word signed number in D100 nicknamed "Dancer" for your dancer position. If you wanted to know that it was a negative, you could use D100:31 which is the sign bit for D100. If you were to nickname D100:31 "Dancer_Neg" you could use this nickname throughout your program and it would always refer to D100:31.

Start Page replacement

You can attach documents to your project to replace the content of the Start Page on a project-by-project basis. This can be a PDF, Excel spreadsheet, JPG, etc. A good example of this is the PID1.dmd example project found in the Examples folder in Do-more! Designer.

Moving Communications links to another PC

You can move your communications links from one PC to another by copying the DmDComm.RST file that is located in the Do-more\Designerx_x\Bin\ folder.

Troubleshooting email with DMLogger

In your Do-more! Designer ladder project, turn $EnableMsgDump bit ON & start Do-more! Logger from Applications window in the Launchpad to troubleshoot email. This will send the server messages to your PC so that you can see exactly where it is getting stuck at.

The DMLogger utility can also be useful for troubleshooting by placing STREAMOUT instructions in your ladder code to send messages to the utility to let you know what is happening at various times.

Importing Program segments

Use the #import mechanism in your Import Program text file so you can modularly create projects from different source files. So MyProject.TXT could contain just 4 lines:

#include “SysConfigXYZZY.txt”

#include “MachineControl.txt”

#include “Library\NonLinearControlLibrary.txt”

#include “Library\CalendarLibrary.txt”