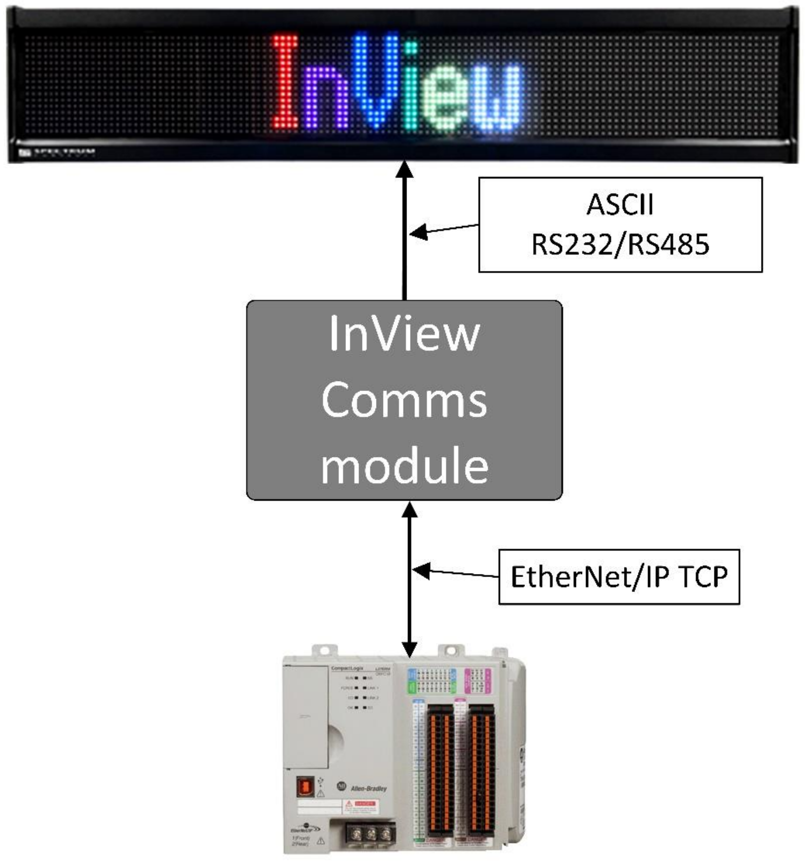
The InView Display's Comms Module provides a simple to use interface for configuring and controlling an InView Display. Configuration is done via a Web Browser and when using Allen Bradley PLC's, control is done in one of two ways, either by using "Classic" mode or "Easy" mode (also called "Easy Tags"). In Classic mode, ladder logic is required to convert a series of integers into strings which are then sent to the InView Display as control strings. When using Easy Tags, a series of INT and SINT tags in the PLC are monitored by the Comms Module which uses the value of these tags to construct the appropriate command strings to send to the InView display. While both methods offer the same level of control, Classic mode requires ladder logic to create the command strings. This makes Easy Tags a much simpler and efficient way to control your InView Display with an Allen Bradley PLC.

This document will discuss the process of using Easy Tags to display primary messages, change the value and control both numeric and alpha numeric variables. Please note that a basic understanding of PLC ladder logic and programming is assumed. For setup and basic configuration of an InView Display and Comms module please refer to the appropriate User's Guide.

InView communications basics



Like most LED displays, the InView Display uses Serial ASCII communications for both programming and control. The InView Comms Module provides a user-friendly interface between the InView Display and the PC/PLC used to configure/control it.

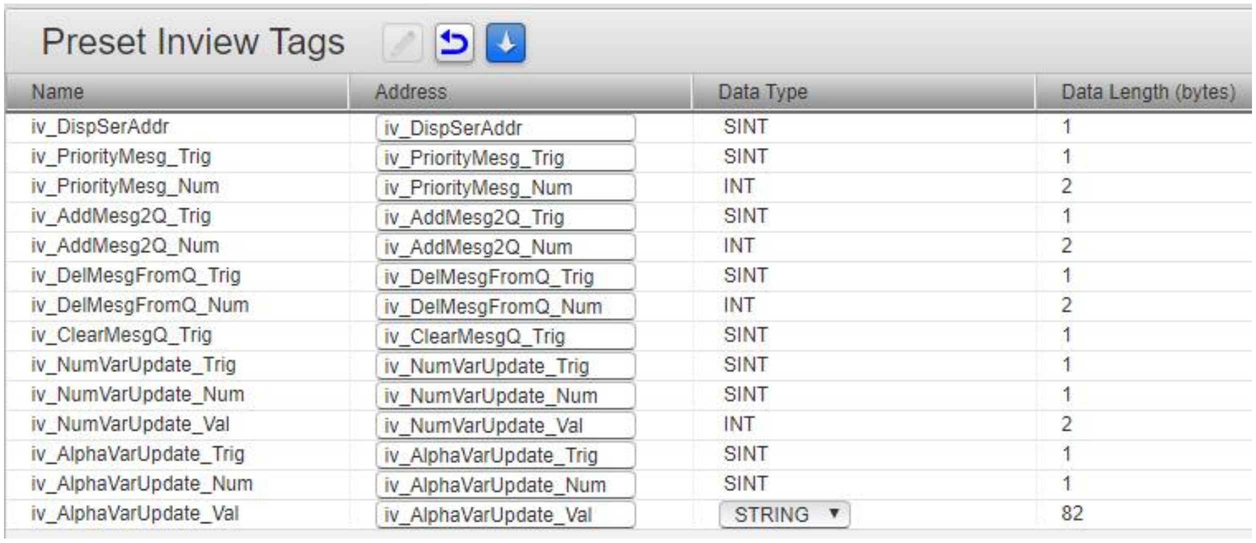
When using Easy Tags, the Comms Module polls a defined series of Tags in the PLC and depending on the value of these Tags, issues commands to the InView Display. Within this process there are three basic functions, (1) displaying Primary Messages, (2) changing and displaying variable data within one or more Primary Messages and (3) controlling the message que feature*. It's worth noting that all messages used in an InView Display are preloaded during the configuration process and they are all "Priority" messages. If the content of a message needs to be created as part of the normal application process, an Alpha Numeric variable must be embedded into a

*Please refer to the Help section of the User Interface (small question mark, upper right hand of the screen) or the User's Guide for more configuration information

Priority Message. It is possible to have a Priority Message with nothing but an Alpha Numeric Variable as content.

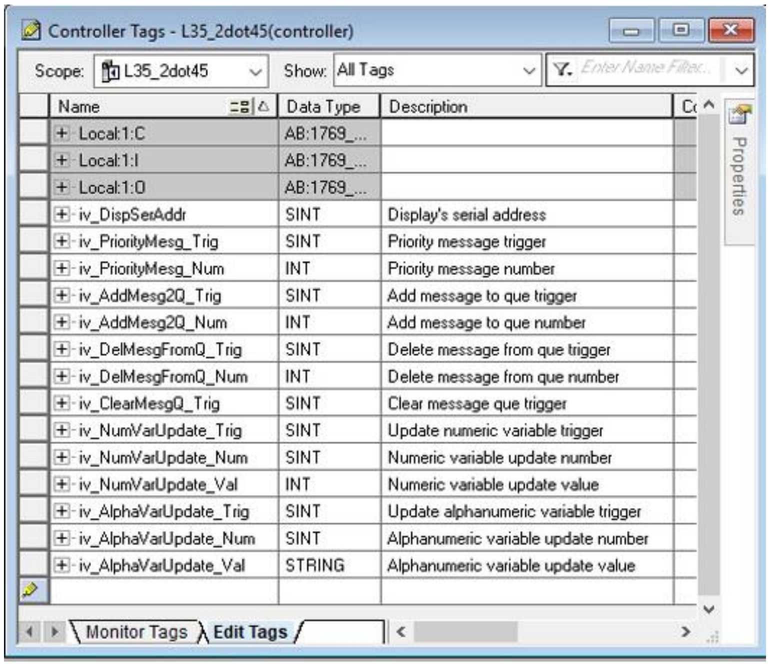
Easy Tags

Easy Tags may vary slightly depending on the type of PLC that will be the Message Server; however, with all types of PLCs, there are 14 Tags used for control. These Tags are pre-programmed in the InView Comms Module and all 14 must also be present in the "Controller Tags" section of the PLC when using a Logix based PLC.



Preset Inview Tags

Name	Address	Data Type	Data Length (bytes)
iv_DispSerAddr	iv_DispSerAddr	SINT	1
iv_PriorityMesg_Trig	iv_PriorityMesg_Trig	SINT	1
iv_PriorityMesg_Num	iv_PriorityMesg_Num	INT	2
iv_AddMesg2Q_Trig	iv_AddMesg2Q_Trig	SINT	1
iv_AddMesg2Q_Num	iv_AddMesg2Q_Num	INT	2
iv_DelMesgFromQ_Trig	iv_DelMesgFromQ_Trig	SINT	1
iv_DelMesgFromQ_Num	iv_DelMesgFromQ_Num	INT	2
iv_ClearMesgQ_Trig	iv_ClearMesgQ_Trig	SINT	1
iv_NumVarUpdate_Trig	iv_NumVarUpdate_Trig	SINT	1
iv_NumVarUpdate_Num	iv_NumVarUpdate_Num	SINT	1
iv_NumVarUpdate_Val	iv_NumVarUpdate_Val	INT	2
iv_AlphaVarUpdate_Trig	iv_AlphaVarUpdate_Trig	SINT	1
iv_AlphaVarUpdate_Num	iv_AlphaVarUpdate_Num	SINT	1
iv_AlphaVarUpdate_Val	iv_AlphaVarUpdate_Val	STRING	82



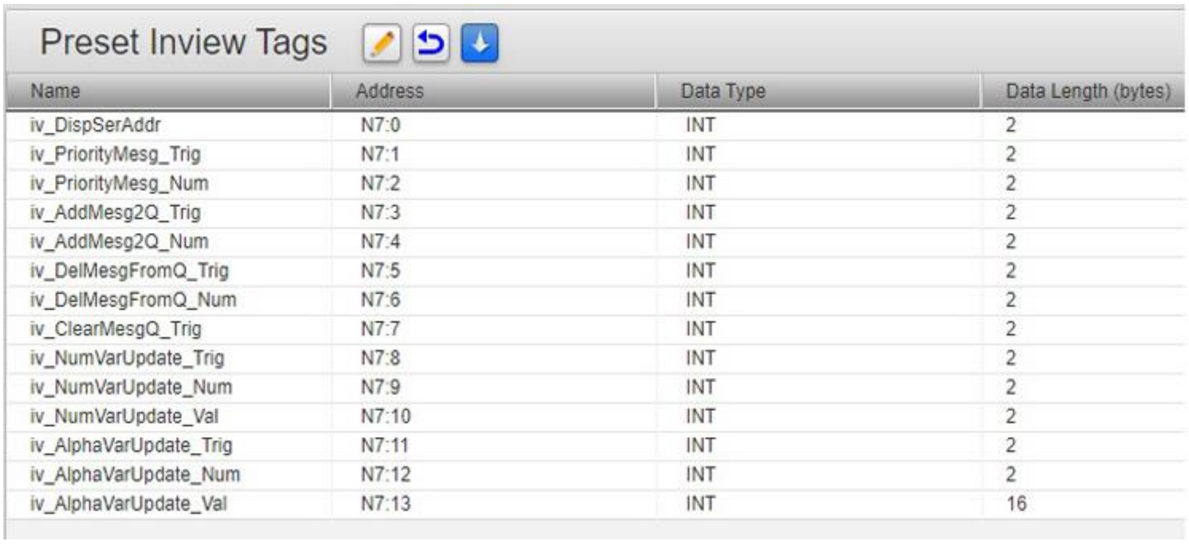
Controller Tags - L35_2dot45(controller)

Scope: L35_2dot45 Show: All Tags

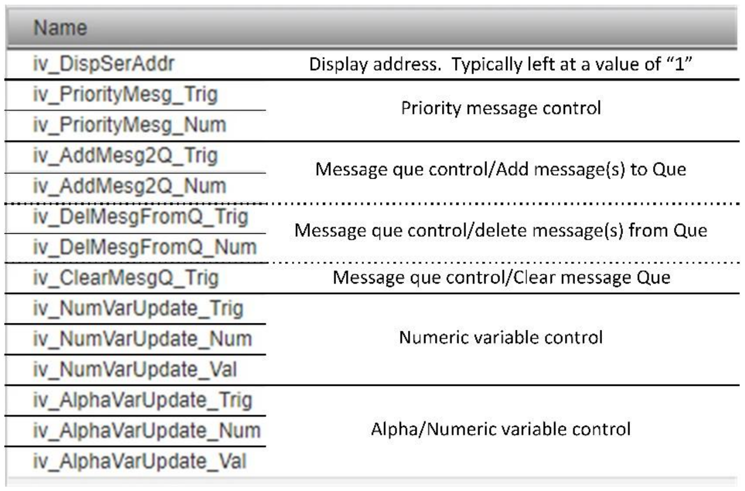
Name	Data Type	Description
Local:1:C	AB:1769_...	
Local:1:I	AB:1769_...	
Local:1:O	AB:1769_...	
iv_DispSerAddr	SINT	Display's serial address
iv_PriorityMesg_Trig	SINT	Priority message trigger
iv_PriorityMesg_Num	INT	Priority message number
iv_AddMesg2Q_Trig	SINT	Add message to que trigger
iv_AddMesg2Q_Num	INT	Add message to que number
iv_DelMesgFromQ_Trig	SINT	Delete message from que trigger
iv_DelMesgFromQ_Num	INT	Delete message from que number
iv_ClearMesgQ_Trig	SINT	Clear message que trigger
iv_NumVarUpdate_Trig	SINT	Update numeric variable trigger
iv_NumVarUpdate_Num	SINT	Numeric variable update number
iv_NumVarUpdate_Val	INT	Numeric variable update value
iv_AlphaVarUpdate_Trig	SINT	Update alphanumeric variable trigger
iv_AlphaVarUpdate_Num	SINT	Alphanumeric variable update number
iv_AlphaVarUpdate_Val	STRING	Alphanumeric variable update value

*Please refer to the Help section of the User Interface (small question mark, upper right hand of the screen) or the User's Guide for more configuration information

When using a SLC or MicroLogix type of PLC the default setup uses “N7” Integer files with a minimum of 21 elements configured. This can be changed to a different file number if it is programmed in the PLC as an “N” (Integer) type file.


Preset Inview Tags

Name	Address	Data Type	Data Length (bytes)
iv_DispSerAddr	N7:0	INT	2
iv_PriorityMesg_Trig	N7:1	INT	2
iv_PriorityMesg_Num	N7:2	INT	2
iv_AddMesg2Q_Trig	N7:3	INT	2
iv_AddMesg2Q_Num	N7:4	INT	2
iv_DelMesgFromQ_Trig	N7:5	INT	2
iv_DelMesgFromQ_Num	N7:6	INT	2
iv_ClearMesgQ_Trig	N7:7	INT	2
iv_NumVarUpdate_Trig	N7:8	INT	2
iv_NumVarUpdate_Num	N7:9	INT	2
iv_NumVarUpdate_Val	N7:10	INT	2
iv_AlphaVarUpdate_Trig	N7:11	INT	2
iv_AlphaVarUpdate_Num	N7:12	INT	2
iv_AlphaVarUpdate_Val	N7:13	INT	16



Name	
iv_DispSerAddr	Display address. Typically left at a value of “1”
iv_PriorityMesg_Trig	Priority message control
iv_PriorityMesg_Num	
iv_AddMesg2Q_Trig	Message que control/Add message(s) to Que
iv_AddMesg2Q_Num	
iv_DelMesgFromQ_Trig	Message que control/delete message(s) from Que
iv_DelMesgFromQ_Num	
iv_ClearMesgQ_Trig	Message que control/Clear message Que
iv_NumVarUpdate_Trig	Numeric variable control
iv_NumVarUpdate_Num	
iv_NumVarUpdate_Val	
iv_AlphaVarUpdate_Trig	Alpha/Numeric variable control
iv_AlphaVarUpdate_Num	
iv_AlphaVarUpdate_Val	

Within the Easy Tag structure, there are five control sections and four types of tags. The control sections are the “Display Address”, “Priority message” control, “Message Que” control and two variable controls, “Numeric” and “Alpha Numeric”. The four types of tags are an “Address” tag, a “Num” (number) tag, a “Trig” (trigger) tag and a “Val” (value) tag. With all five control sections there are a total of 14 tags and even if the application will only be cycling the Priority messages, all 14 tags must still be present in the PLC and must be an exact match between the Comms Module and the PLC tags. Because of this, the recommended method of configuring the tags in a Logix PLC is to use the Comms Module’s “export” feature. This feature allows for the tags within the Comms Module to be exported in a file format (.csv) that can be imported into the Logix based PLC. In the applications using SLC’s and MicroLogix type PLC’s, the Comms Module’s tag structure mimics that of the “N7” type integers. It is important to note that when using a SLC or MicroLogix as the message server, the “N7” file data table must have at least 21 elements in order for the Comms Module to make a connection with the PLC. It is also worth noting that if the PLC is already using the “N7” Integers, it is possible to change the Comms Modules configuration to use another Integer number; however, there must still be at least 21 elements.

*Please refer to the Help section of the User Interface (small question mark, upper right hand of the screen) or the User’s Guide for more configuration information

Types of tags

Address: The Address tag is used to define which sign the control command is being issued to when there are multiple InView signs connected to one Comms Module. In most applications there is only one InView sign per Comms Module so an address of "1" is typically used. Because of this, the Address tag within the PLC can be configured with a value of "1" and never need to be used in the PLC's ladder logic. However, best practice is to monitor the value of the Address tag within the PLC and change it to a value of "1" if for some reason it is changed to something else.

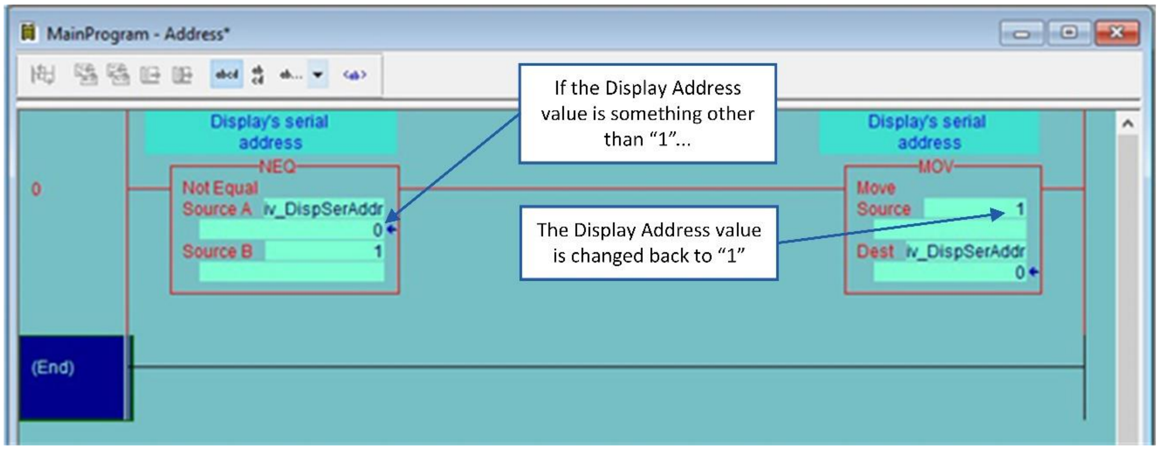
Num (number): The "Num" or "Number" tag defines which priority message is being controlled or which variables within a priority message are being controlled.

Val (value): The "Val" or "Value" tag is used in both Numeric Variables as well as in Alpha-Numeric Variables and defines the value of the variable part of a message.

Trig (trigger): The "Trig" or "Trigger" tag is used to execute the command. This tag operates slightly different from the others in that its value is not monitored but rather its changing value is used to trigger the command. In other words, the trigger tag does not have to be set to a specific value to trigger a command it simply has to change state (I.E. change from a "0" to a "1" or a "5"). This is important to understand for a couple of reasons, first, it is probably the most commonly overlooked item when trying to execute a command such as changing which priority message to display. Second, it makes the PLC programming simpler to do with regards to executing commands to the Comms Module (this will be covered later in this document).

Using Easy Tags to control your sign

Address:
The first step in controlling an InView Display is setting the address tag. This can be done by simply setting the address tag, "iv_DispSerAddr" to whatever the serial port address on the InView sign itself is (typically this address is set to "1") or you can create a simple routine in ladder logic to set the address automatically.


MainProgram - Address*

If the Display Address value is something other than "1"...

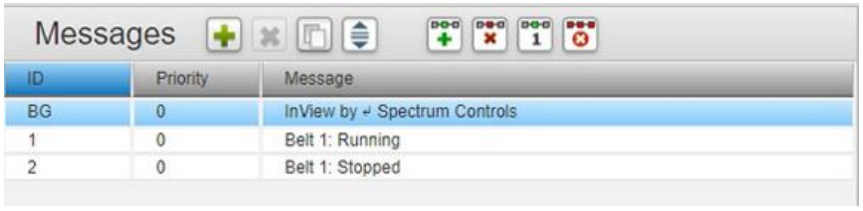
The Display Address value is changed back to "1"

```
0   Display's serial address              Display's serial address
    NEQ                                   MOV
    Not Equal                             Move
    Source A  iv_DispSerAddr              Source         1
                           0              Dest  iv_DispSerAddr
    Source B               1                             0
(End)
```

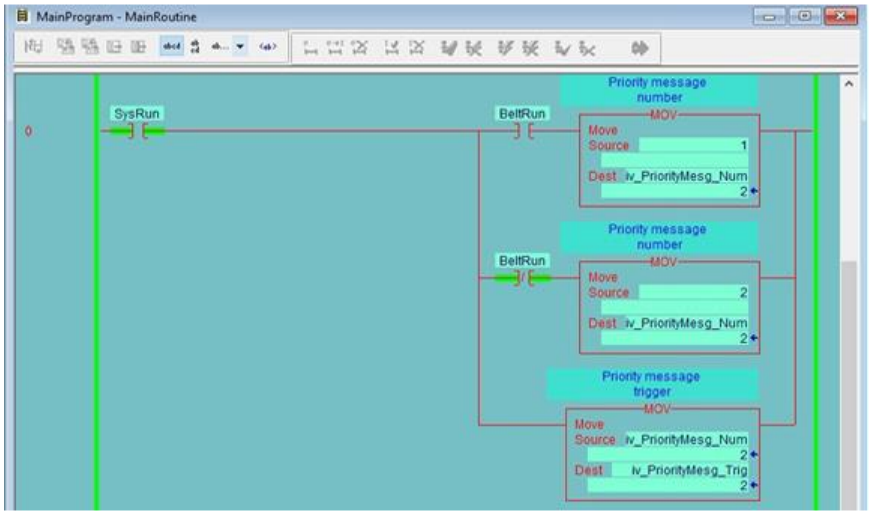
*Please refer to the Help section of the User Interface (small question mark, upper right hand of the screen) or the User's Guide for more configuration information

Priority Messages:
As previously mentioned, all messages that will be displayed in any given application must be pre-programed and loaded into an InView Display. These are called “Priority Messages” and are all given a unique ID number which is used to display the message. To display a message, the value of “iv_PriorityMesg_Num” (message number tag) is set to the ID number of the message and “iv_PriorityMesg_Trig” (trigger message tag) is triggered by changing its value. Because the trigger is activated when the value of the trigger tag changes (rather than what it changes to), the simplest way to trigger a message is to move the value of the message number tag to the trigger message tag and then only update the message number tag when you are ready for the message to be displayed.

To illustrate this, let’s create an example application where the status of a conveyor belt called “Belt 1” needs to be displayed. Because the options are “Running” and “Stopped”, two messages are needed with the “Running” message having an ID of “1” and the “Stopped” message having an ID of “2”.



ID	Priority	Message
BG	0	InView by ↵ Spectrum Controls
1	0	Belt 1: Running
2	0	Belt 1: Stopped



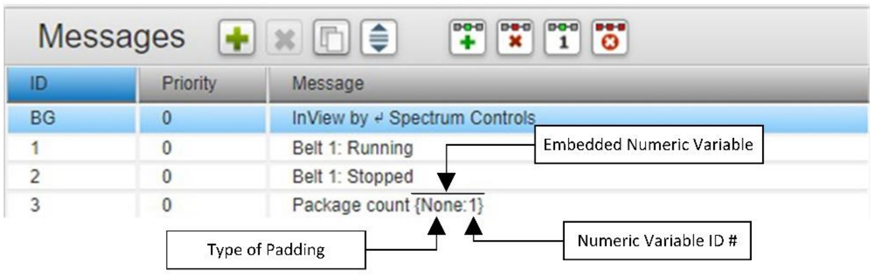
```
Priority message number
MOV
Move
Source 1
Dest iv_PriorityMesg_Num
2

Priority message number
MOV
Move
Source 2
Dest iv_PriorityMesg_Num
2

Priority message trigger
MOV
Move
Source iv_PriorityMesg_Num
2
Dest iv_PriorityMesg_Trig
2
```

In the PLC’s ladder logic, when the bit “BeltRun” is high, a value of “1” is moved to the message number tag and when it is low a value of “2” is moved to the message number tag. The last operation is moving the message number tag value to the message trigger tag which triggers either Priority Message 1, “Belt 1: Running” or Priority Message 2, “Belt 1: Stopped” depending on the message number tag value. Please note that regardless of how the message trigger tag is changed, this must be done after the message number tag’s value is set.

*Please refer to the Help section of the User Interface (small question mark, upper right hand of the screen) or the User’s Guide for more configuration information

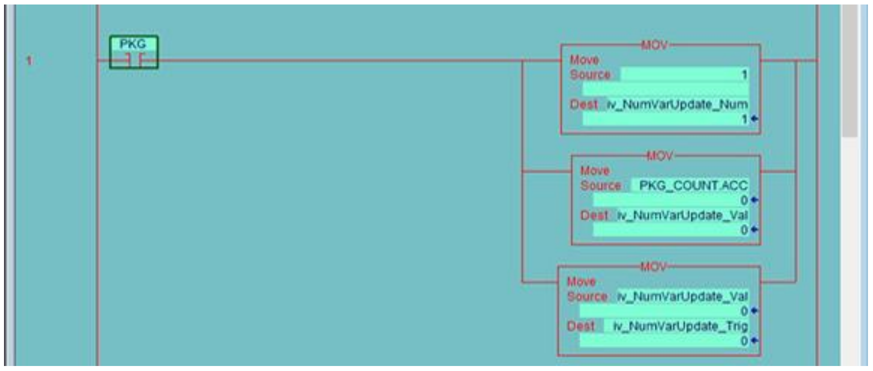
Numeric Variables:
Now let's add a Numeric Variable to the example. Packages on the conveyor belt are being counted and when needed, the number counted needs to be displayed.

First, we need to create a Priority Message with a Numeric Variable embedded in it. When you create a Numeric Variable in a Priority Message*, you assign it an ID number and the type of padding, None, Spaces or Zeros.



ID	Priority	Message
BG	0	InView by ↵ Spectrum Controls
1	0	Belt 1: Running
2	0	Belt 1: Stopped
3	0	Package count {None:1}

Next, to display the Numeric Variable, the Priority Message with the embedded Numeric Variable is triggered and once that is done the variable value can be updated. To do this there are three tags that are used, the "iv_NumVarUpdate_Num" tag is used to identify the variable that is embedded within the Priority Message, the "iv_NumVarUpdate_Val" tag is the variable value and "iv_NumVarUpdate_Trig" is the trigger tag.

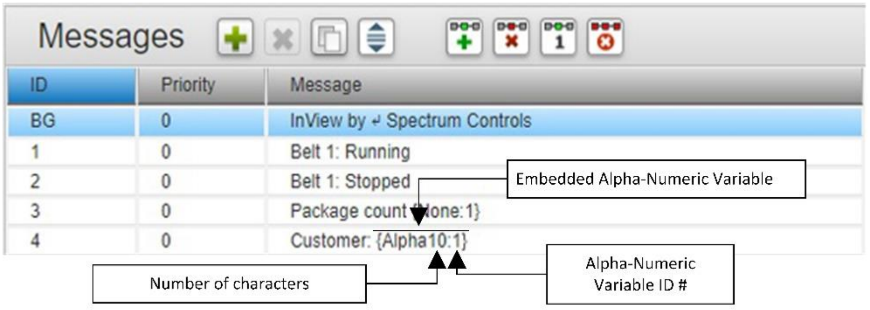


In the example ladder logic, when the bit "PKG" is high, a value of "1" is moved to the PLC tag for the Numeric Variable ID (iv_NumVarUpdate_Num) after which the value is moved to PLC's variable value tag (iv_NumVarUpdate_Val). Once the ID number and value is set the trigger tag (iv_NumVarUpdate_Trig) is updated which updates the variable value in the message.

Alpha Numeric Variables:
The last part of this application example will be adding an Alpha Numeric Variable. Because Alpha Numeric Variables use Strings rather than integers for the variable value, the process of executing an Alpha Numeric Variable is slightly different; however, the basic principle is the same. We first identify which Alpha Numeric Variable is to be updated, set the value and then trigger the update.

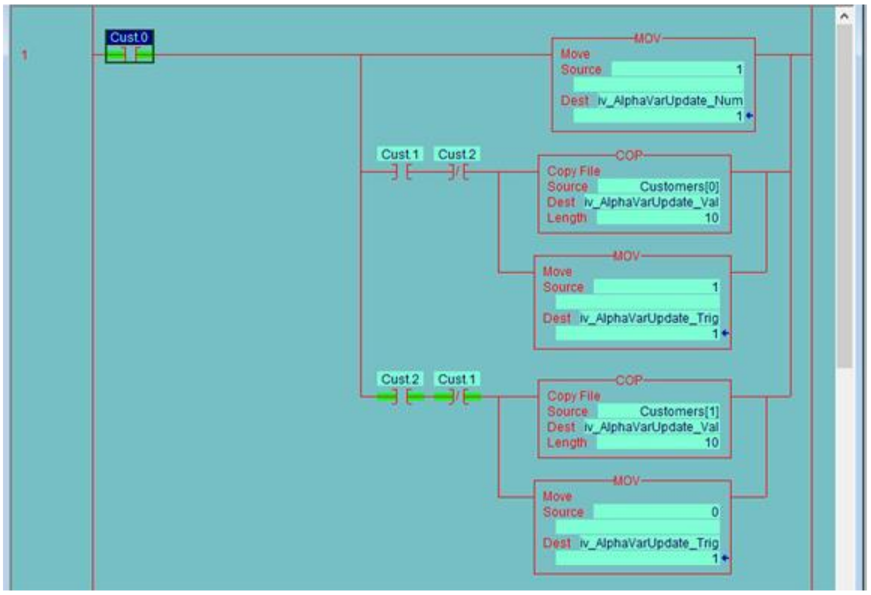
*Please refer to the Help section of the User Interface (small question mark, upper right hand of the screen) or the User's Guide for more configuration information

In the application example we are going to add a message that identifies a customer. The Priority Message setup is the same as with the Numeric Variable message but when creating the message*, instead of selecting “Numeric”, we select “Alphanumeric” and set the variable ID and max text Length.
Now that we have the Priority Message with an embedded Alphanumeric variable, we are ready to execute the ladder logic.



Messages

ID	Priority	Message
BG	0	InView by ↵ Spectrum Controls
1	0	Belt 1: Running
2	0	Belt 1: Stopped
3	0	Package count {None:1}
4	0	Customer: {Alpha10:1}



```
1  Cust 0 ── MOV  Move
                  Source 1
                  Dest iv_AlphaVarUpdate_Num  1
   Cust 1 ─ Cust 2 (/) ── COP  Copy File
                              Source Customers[0]
                              Dest iv_AlphaVarUpdate_Val
                              Length 10
                         ── MOV  Move
                              Source 1
                              Dest iv_AlphaVarUpdate_Trig  1
   Cust 2 ─ Cust 1 (/) ── COP  Copy File
                              Source Customers[1]
                              Dest iv_AlphaVarUpdate_Val
                              Length 10
                         ── MOV  Move
                              Source 0
                              Dest iv_AlphaVarUpdate_Trig  1
```

First, as with the Numeric Variable, we activate the Priority message containing the Alphanumeric Variable then we can update the variable. In the example ladder logic, when bit “Cust.0” is high, a value of “1” is moved to the PLC tag for the Alpha Numeric Variable ID (iv_AlphaVarUpdate_Num). When the bit “Cust.1” is high and “Cust.2 is low, a Copy instruction is used to move the value of the String file “Customer[0]” to the variable value tag (iv_AlphaVarUpdate_Var) and a “1” is moved to the trigger tag (iv_AlphaVarUpdate_Trig) which updates the variable in the message. When the bit “Cust.1” is low and “Cust.2” is high, a copy instruction is used to move the value of the String file “Customer[1]” to the variable value tag and a “0” is moved to the trigger tag which updates the variable in the message.

Conclusion:

The classic method of controlling an InView Display is a tried and true method; however, it requires extensive PLC programming to convert Integers into strings and assembling the strings into the correct commands that can be understood by the InView Display. Easy Tags takes all of that PLC programming and incorporates it into the InView Comms Module, making what little PLC programming is left simple and efficient. The examples shown above illustrate just how simple it is to add or update an InView Display for any automation application.

*Please refer to the Help section of the User Interface (small question mark, upper right hand of the screen) or the User’s Guide for more configuration information