

# Panelview

Using Panelbuilder32 software

**EthernetSupport.com**

*“Customized Automation Training”*

Copyright (c) 1999 Ricky Bryce.

Permission is granted to copy, distribute verbatim copies of this document, commercially or non-commercially with no front cover texts, or back cover texts. Changing this document is not allowed.

Disclaimer:

This document is written in the hope that you can utilize for your own education to gain knowledge of PLC systems (should you decide to utilize this document).

Although I believe the information in this document to be accurate, it is YOUR responsibility to verify this information before implementing it in any way, especially when damage to personnel or equipment could result.

By continuing to read this document, you agree to hold no one who writes, modifies, or distributes this document liable in any way (even negligence).

Due to the wide variety of plant applications, some of the examples in this document may be prohibited at your location, or could cause damage to equipment, or harm personnel.

About the Author:

I'm Ricky Bryce, and I'm employed as an instructor of Automation Training, Inc (ATI) since April, 2004. This document is a collection of texts and graphics I've put together over the past few years, and has been distributed under the GFDL since 1999.

I hope you get much use out of it, and I would like your feedback as to how this document can be improved.

As a supplement to this document, I would like to invite you to my website at <http://www.LearnAutomation.com>. I'm in the process of uploading documentation and videos that will further help you with problems or questions you have with Allen Bradley processors.

You are also invited to visit my employers site at <http://www.atifortraining.com>.

If you are interested in on site training, please contact my employer at **(866) 573-9849**

"Human Knowledge Belongs to Everyone"

# Table of Contents

Panelview PreTest.....	6
Configuration Screen.....	9
Creating a new RSLogix 500 Project.....	14
Creating a New Project in Panelbuilder 32.....	19
Configuring the DF1 Driver.....	23
Text Object.....	26
Remote I/O Considerations for the SLC.....	28
Pushbutton Objects.....	29
Multistate Indicators.....	33
Navigation.....	36
Numeric Data Display.....	40
Numeric Entry.....	42
Alarms.....	44
Message Object.....	49
Global Objects.....	52
Graphics.....	55
Screen List Selector.....	61

Name \_\_\_\_\_

Date \_\_\_\_\_

Score \_\_\_\_\_

## ***Panelview PreTest***

- 1) What type of objects on a Panelview Screen would be considered **DISCRETE INPUTS** to the PLC?
  
- 2) What type of objects on a Panelview Screen would be considered **DISCRETE OUTPUTS** from the PLC?
  
- 3) What type of objects on a Panelview Screen would be considered **ANALOG INPUTS** to the PLC?
  
- 4) What type of objects on a Panelview Screen would be considered **ANALOG OUTPUTS** from the PLC?
  
- 5) What is the definition of a Bit of memory?
  
- 6) What is the definition of a Word of memory?
  
- 7) Anything added to the screen of a Panelview Application is called an \_\_\_\_\_?
  
- 8) What feature of the Panelview would you configure to inform an operator if a tag exceeds a certain preset value?

- 9) What is the term used for transferring a Panelview application from your computer to the Panelview terminal?
- 10) What are two uses for the serial port (RS232) on the Panelview Terminal?
- 11) What communication protocols can be used to interface a Panelview terminal with a PLC?
- 12) What is the purpose of Data Tables in the PLC?
- 13) What is the purpose of Program Files in the PLC?
- 14) While interfacing with the PLC, which of these files can the Panelview access?
- 15) What type of object would you use to place a label on a Panelview Screen?
- 16) What is a Global Object?
- 17) What type of object would you use to allow the operator to change the value of a bit address in the PLC (between 1 and 0)?
- 18) What object would you place on the screen to allow the operator to enter a numeric value (such as to change the speed of a motor, or the preset on a timer)?

- 19) What object would you use to reflect the state of an output (running or not running)?
  
- 20) What object would you place on a Panelview screen to reflect the value of an analog signal in the PLC such as the accumulated value of a timer or counter?
  
- 21) How would you access the configuration screen on a standard Panelview terminal?
  
- 22) What information can be found on the “Terminal Info” Screen?
  
- 23) What has to be the same about every device on the same Data Highway Plus or Remote I/O Network network?
  
- 24) What has to be different about every device on the Data Highway Plus or Remote I/O Network?
  
- 25) What is the maximum allowed distance for a DH+ or Remote I/O Network?



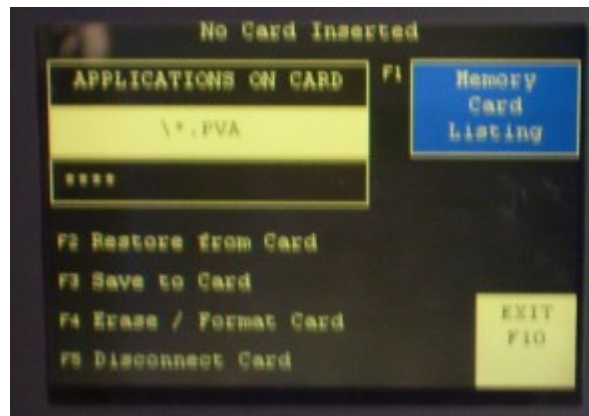
## *Configuration Screen*

To access the configuration screen, you can press the right and left keys simultaneously on the Panelview terminal for keypad units. On the Touch Screen units, you will notice a square on the display screen when you power up. Hold the touch cell on top of the square while the unit powers up. It is also possible for the programmer to add a button to a display screen to enter the Panelview Configuration.

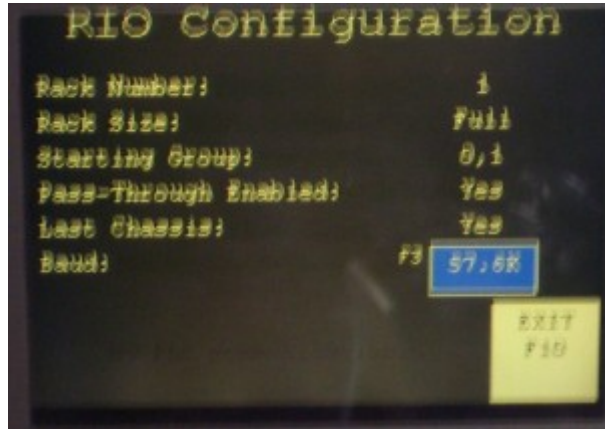
- 1) Press the Right and left arrow keys simultaneously on your Panelview terminal to access the configuration screen:



- 2) Let's look at the first item on the list: Memory Card. If this Panelview terminal was equipped with a memory card, we would see various Panelview applications that we could load into our terminal. Press the F10 key to Exit back to the Main Menu.



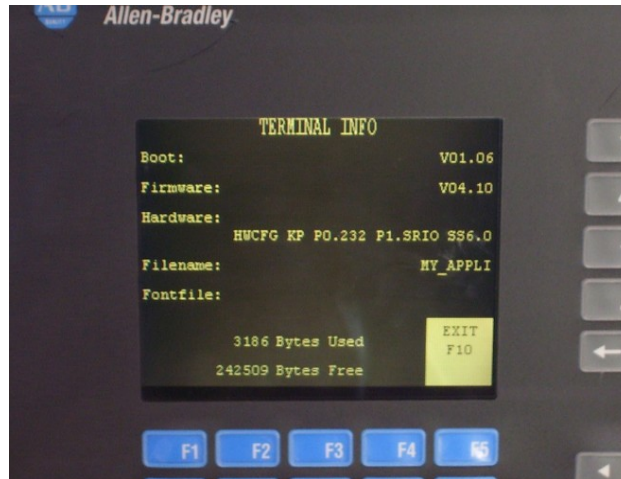
- 3) Next Let's look at Communication Setup. This screen displays information about the communication configuration used by the Panelview Terminal. This information includes the Rack, Starting Group, Baud Rate, and Size.



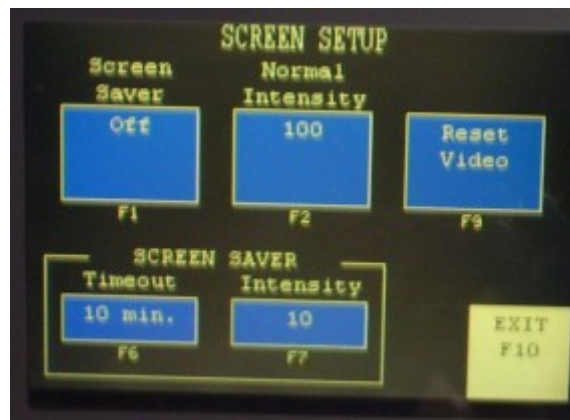
- 4) Let's look at the next item on the Configuration Screen List: Preset Operations. On this screen we can tell the Panelview to power up with the preset values that were set in Panelbuilder32 software, or revert to the last known states. A few other options can also be set such as the repeat rate, and the repeat delay. Press F10 to exit back to the main menu.



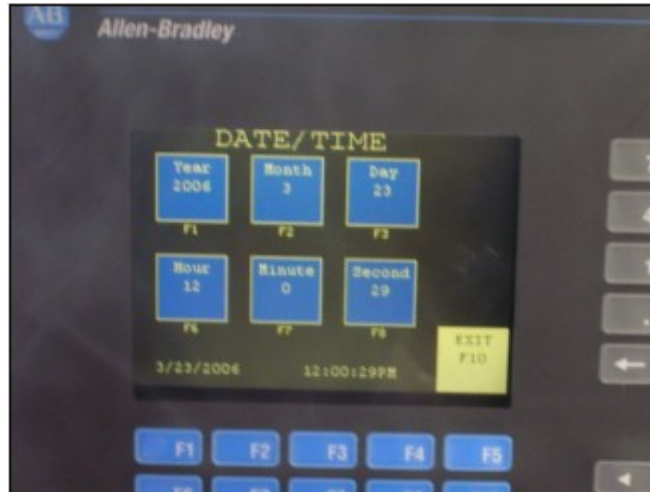
- 5) Now let's look at "Terminal Information". This screen displays various information about the Panelview Terminal such as the firmware revision, application file name, and memory usage. Press F10 to exit back to the Main Menu.



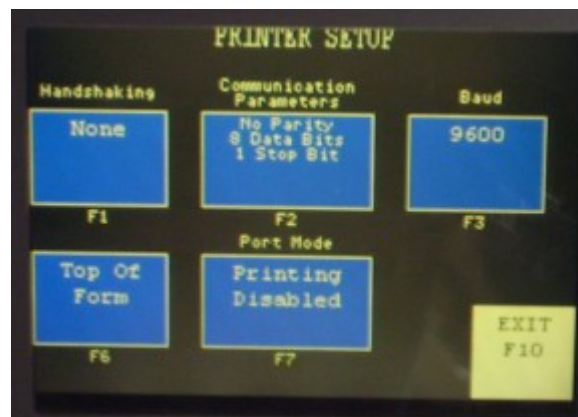
- 6) Next on our list is "Screen Setup". This screen allows us to adjust the characteristics of our display screen such as brightness, contrast, and screen saver setup. Again, press the F10 key to exit back to the main configuration screen when you are finished exploring these parameters.



- 7) On the next tab, you can adjust the Panelview date and time. This information is useful on the alarm screen and alarm reports that are sent to the printer.



- 8) The last item on the list is Printer Setup. Please be aware that in order to upload and download to the Panelview terminal through the RS232 port, the printer must be disabled. If the printer is enabled, you can use an RS232 printer for printing various alarm messages.



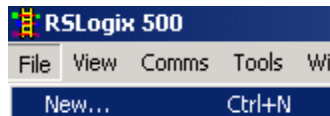
## Creating a new RSLogix 500 Project

In this section, we will create a new RSLogix 500 project. This will be the project we use throughout the rest of the course for use with our Panelview.

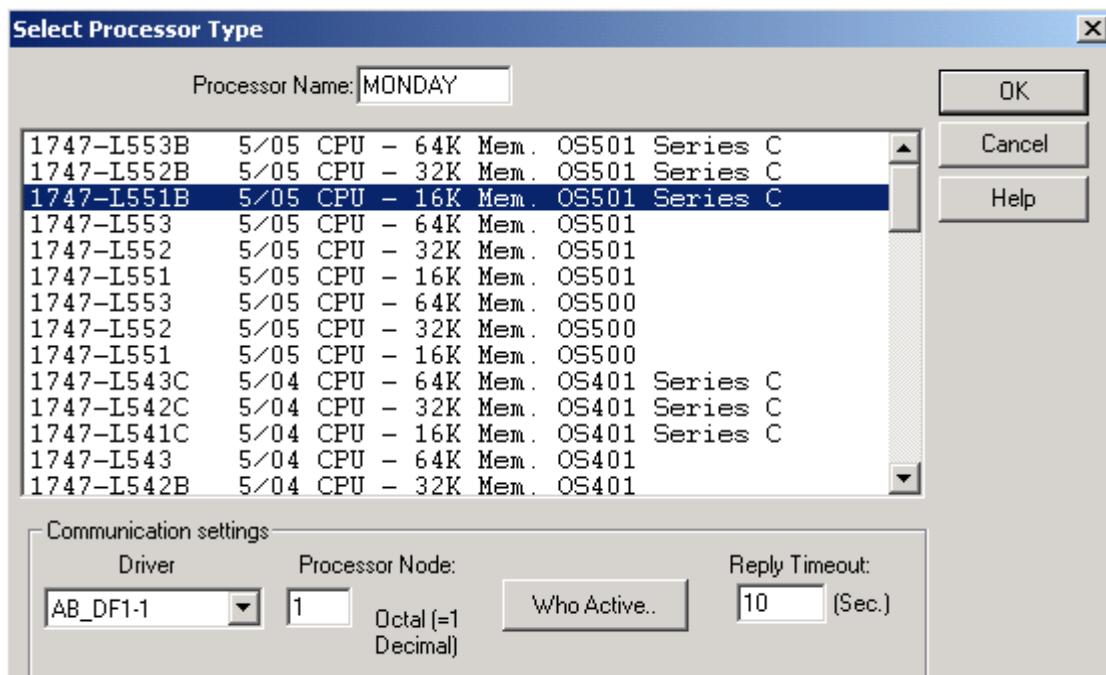
- 1) Open RSLogix 500. To open RSLogix 500, click Start | Programs | Rockwell Software | Rslogix 500 | RSLogix 500 English.



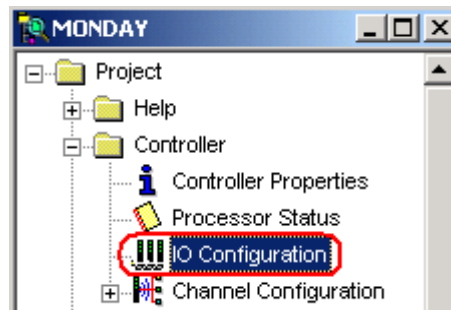
- 2) Click File | New from the menu bar to create a new RSLogix 500 project.



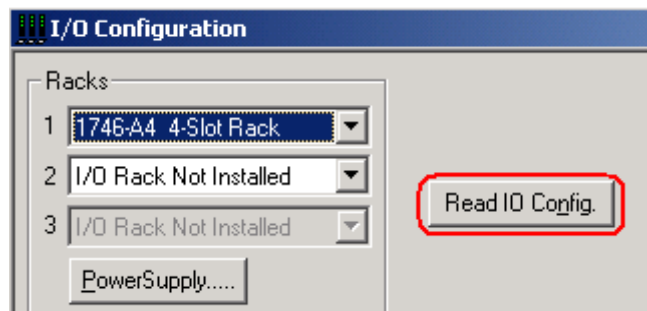
- 3) We will name this project "Monday". Be sure to choose the proper processor for your workstation, and use the DF1 driver in RSLinx to build a path to the processor at Node 1 (as shown below) Press OK when finished.



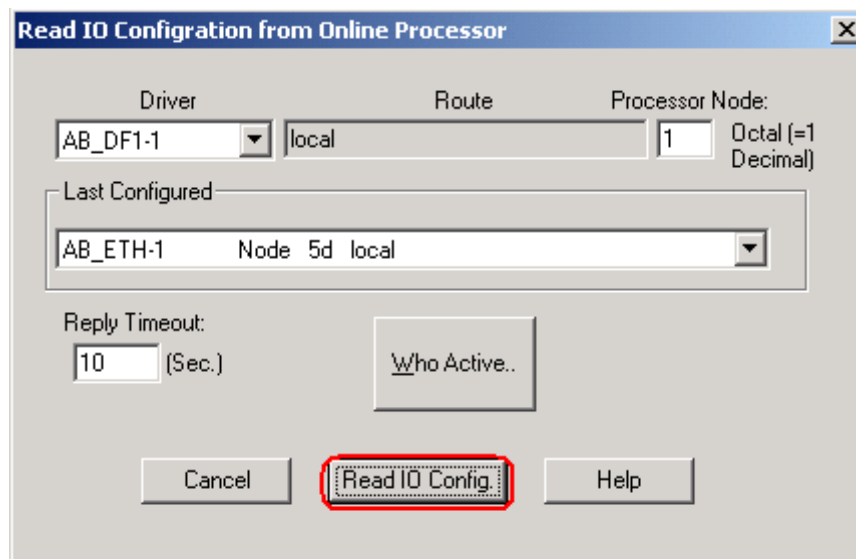
- 4) Next we need to go into the I/O Configuration for this project, and tell the processor what modules to expect in the local chassis.



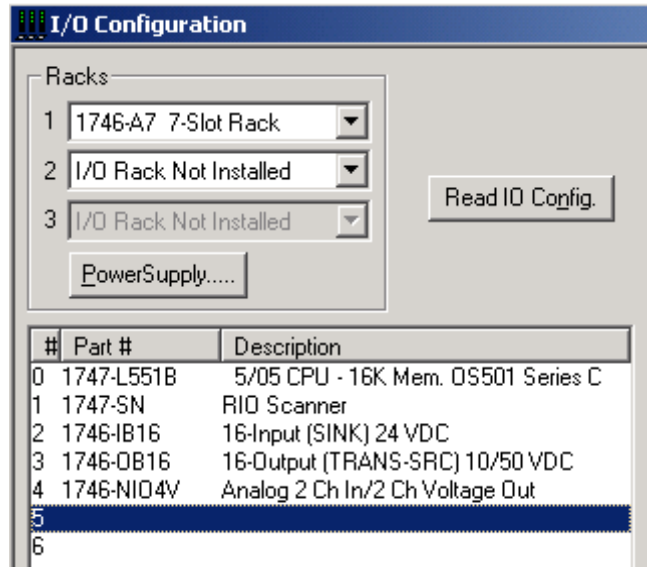
- 5) We could drag and drop modules from the list of available module types, but if you have the 5/03 or higher processor, you can choose to read the I/O modules automatically.



- 6) Next you will be asked to verify your communication path. If everything is OK, just read the I/O Configuration.



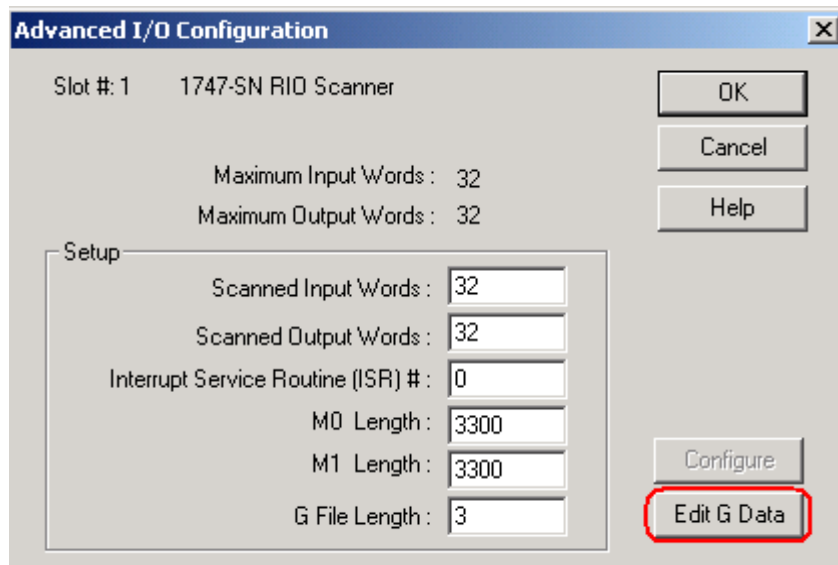
7) You will notice the I/O Configuration list was populated automatically.



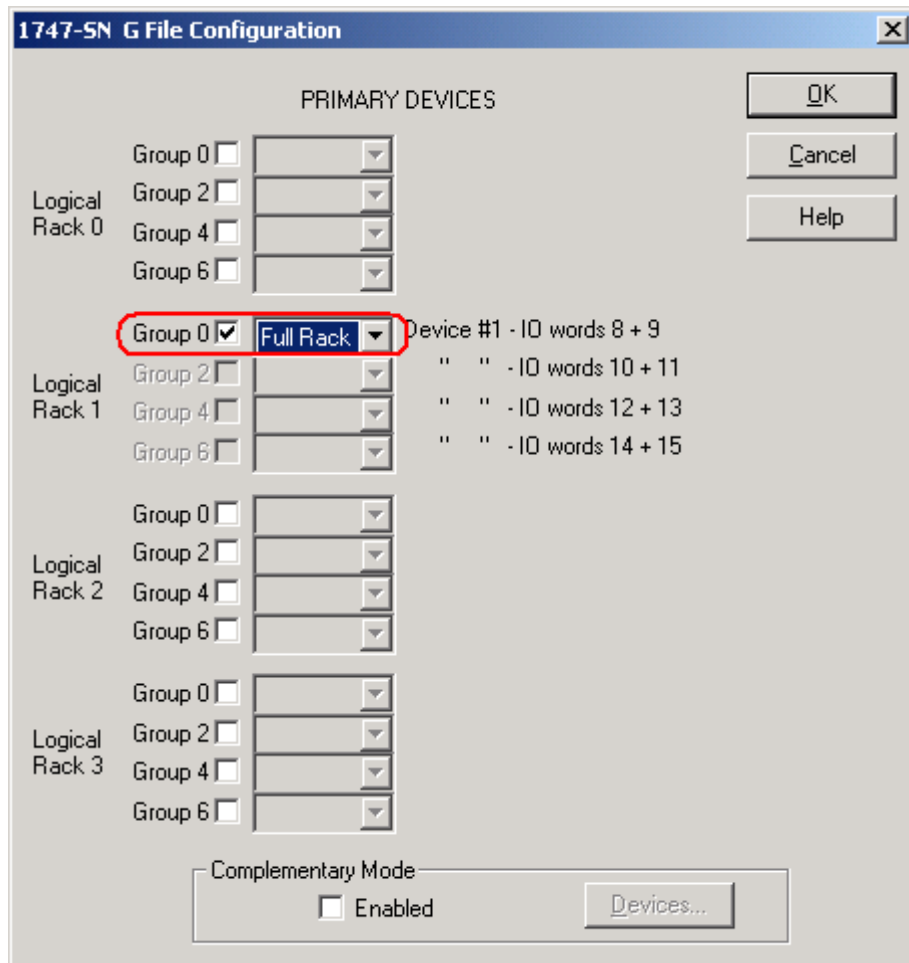
- 8) We need to take this one step further since we also have I/O on the scanner module (The Panelview). If you go into the Communication Setup menu for the Panelview Terminal, you will see it was configured for Rack 1, Starting Group 0, Full Rack... We need to tell the scanner to scan this device.
- 9) Double click the Scanner module in slot 1 (or highlight the module, and choose Advanced Configuration).



10) On the Advanced Configuration screen, click “Edit G Data” (G is the middle letter of the word “confiGure”)



11) Check Rack 1, Starting Module Group 0, and make this chassis a Full Rack as shown:



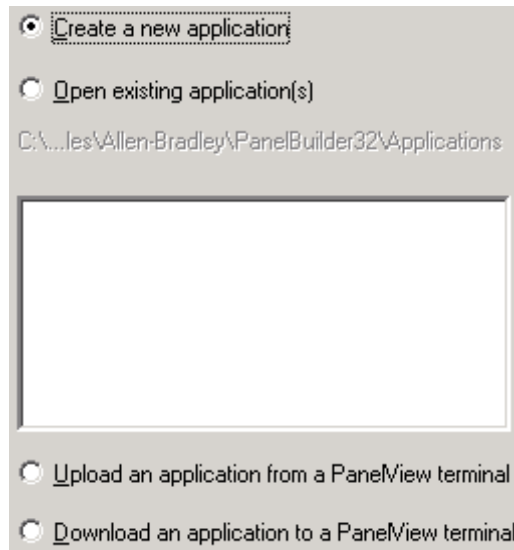
12) Download your work and verify communications are working properly to the Panelview terminal. If so, the Comm light will be solid green on the scanner module in slot 1, and the Comms light will be solid in the Panelview configuration menu.

## *Creating a New Project in Panelbuilder 32*

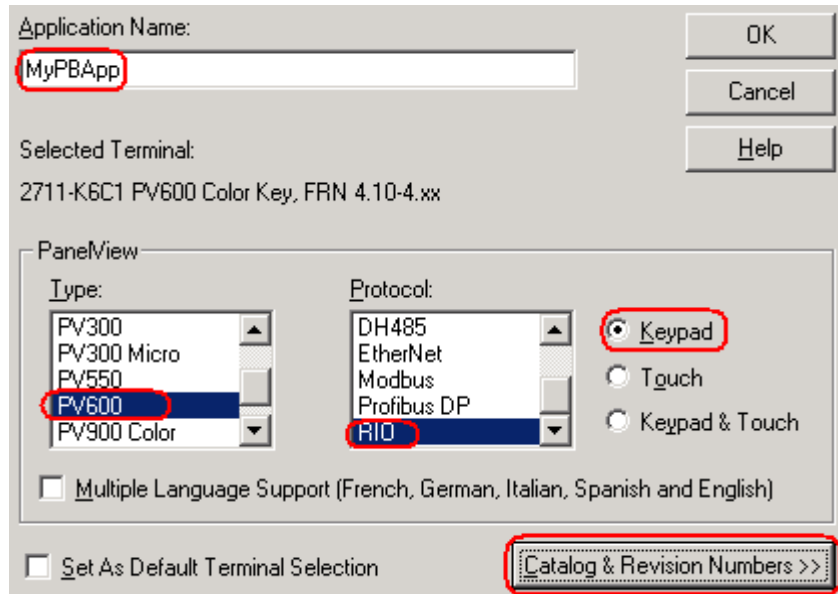
- 1) Open **Panelbuilder32** software. If you don't have a shortcut on your desktop, you can access Panelbuilder32 by clicking **Start | Programs | Panelbuilder32 | Panelbuilder32**.



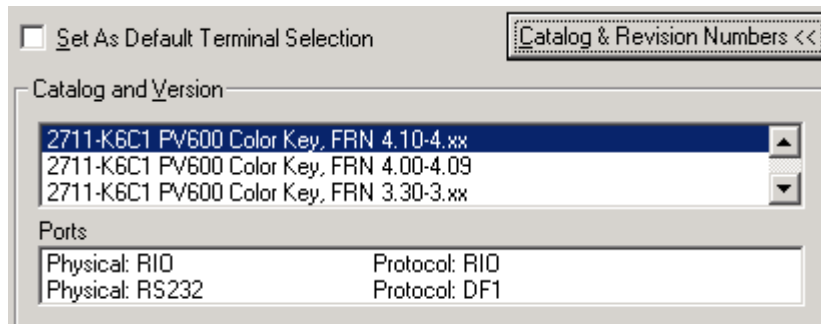
- 2) Next, we are going to **create a new application**. Notice the other options available on this screen. You could also open an existing application from a file on the hard drive, upload, or download from a Panelview terminal. Press **OK** after making your selection.



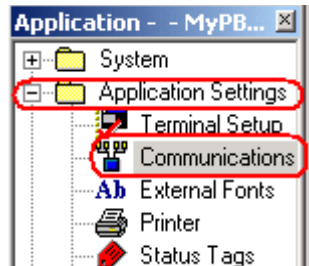
- 3) Next, we have the “create new application” dialog screen. Let's name this application, “MyPBApp”. You have a Panelview 600 terminal that is Keypad only, and runs the RIO (Remote I/O) Protocol. Next, click the button “Catalog & Revision Numbers. We must specify what firmware is in your Panelview Terminal.



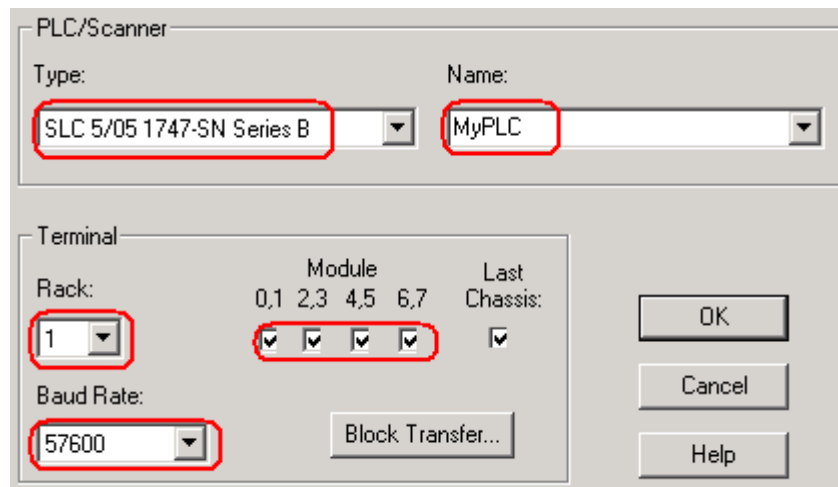
- 4) Your firmware revision may be printed on the back of the Panelview Terminal itself, or you can get the firmware information from “Terminal Info” on the Configuration screen. Press OK when you are finished.



- 5) Since this is a Remote I/O Panelview, we must configure the Panelview for Remote I/O Communication. This means we have to assign the Panelview the following Parameters:
  1. Baud Rate
  2. Rack
  3. Starting Module Group
  4. Size (1/4, 1/2, 3/4, or FULL rack)
  
- 6) To get started, let's open the **Application Settings** folder in your application tree, then double-click “communications”



- 7) Choose the type of PLC you have. For the example below, we have an SLC 5/05 with a series B scanner module. We will name the processor “MyPLC”. This Panelview will be configured for Rack 1. We will start at Module group 0, and will utilize all 8 module groups within rack 1, so our size will be considered a FULL Rack. The baud rate is 57.6K. We'll deal with block transfers at a later time. Just press OK.



8) Congratulations! You are ready to start building your screens. Just some quick review questions first:

1. What does the term UPLOAD mean when referring to a Panelview application?
2. What does the term DOWNLOAD mean?
3. What are the 4 things we have to know about any device that runs on remote I/O (including the Panelview)
4. How many GROUPS are in each rack on Remote I/O Addressing?

## Configuring the DF1 Driver in RSLinx

The DF1 Driver is used for point to point communication over RS232 between a COM port on a PC, and the serial port (Channel 0) of a processor. The following steps will take you through a sample configuration of the DF1 RS232 driver.

- 1) Open RSLinx Communication Server. If there is no short-cut on the desktop, you can access RSLinx by clicking Start | Programs | Rockwell Software | RSLinx | RSLinx



- 2) Click 'Communication' on the menubar, then choose 'Configure Drivers'.

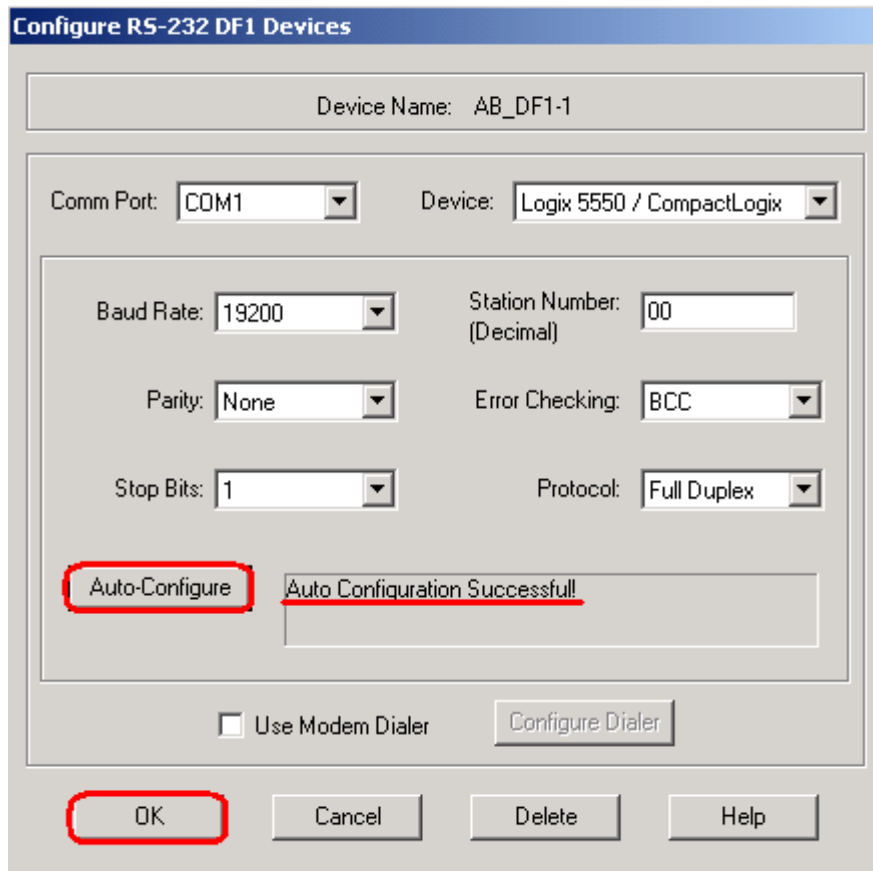


- 3) From the Available Driver Types pull down menu, choose 'RS232 DF1 Devices', then press the ADD NEW button.



- 4) For this example, the name can be left at default. Press OK.

- 5) Although the communication parameters can be entered manually, if you are currently connected to the processor, just hit the 'AutoConfigure' button. RSLinx will hit the processor with different baud rates, and different settings, until it finds a setting it gets a response on. When this happens, you will get a message that the autoconfiguration was successful. Press OK when finished.

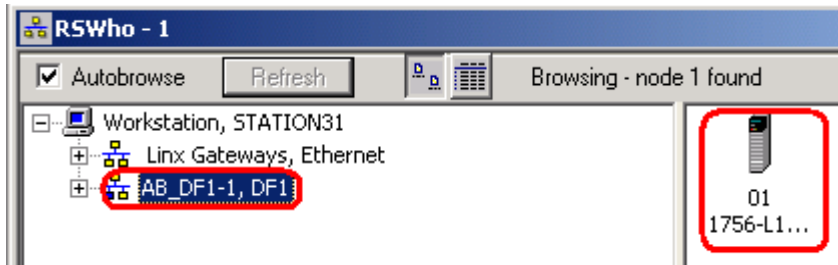


- 6) You will see the driver is now running. Close the "Configure Drivers" screen.



7) est your drivers, click the RSWho icon in the toolbar of RSLinx.

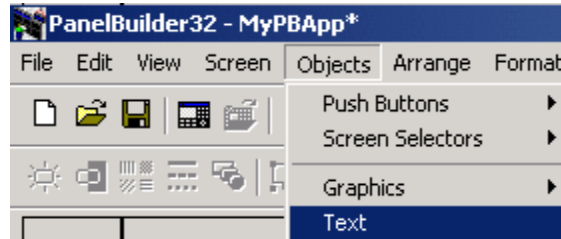
8) Click the DF1 driver (the one you just cofnigured) on the left side of your screen. The devices you are communicating with will appear on the right.



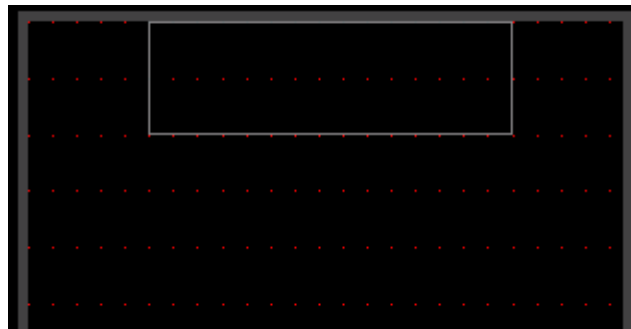
To go online, you must go to RSLogix at this point.

## Text Object

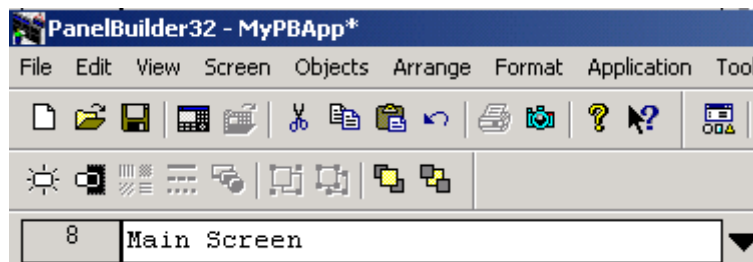
- 1) Labels that we add to screens are called *Text Objects*. To add text to your screen, be sure your screen has focus, and click **objects** from the menu bar, and choose “**text**”.



- 2) Next, (Now that you have the text object) click on the upper left area of where you want the text to start, and drag your mouse button to the lower right area of where you expect your text to end (you can move and resize this later)



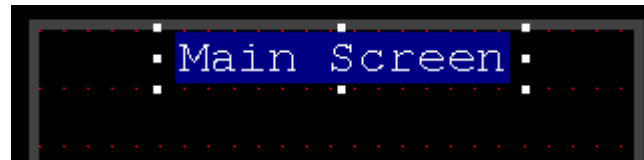
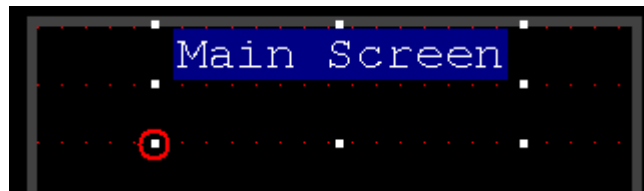
- 3) In your text entry box, type “**Main Screen**”



- 4) Use the text tools to size and adjust your text.



- 5) Next, to save space on your screen for other objects, grab the handles on the text object, and resize so it's not any larger than it needs to be to hold the text.



- 6) Next, you can download to your terminal to see the results of the change you just made to your screen.

## ***Remote I/O Considerations for the SLC***

When running remote I/O on the SLC, there are a few considerations to think about.

- The SLC processor can only see I/O in its local chassis, or an extended local chassis. Therefore all Remote I/O appears to be in the slot number the 1747-SN module is located in.
- The 1747-SN module converts the old PLC-5 style addressing into an address the SLC can understand.

The next few pages will explain how this takes place.

### **Review of PLC-5 Style addressing:**

Let's understand some terminology from the old PLC-5 style addressing scheme.

- **Bit:** The smallest unit of information in the PLC. The only allowed states are On or Off.
- **Word:** 16 Bits
- **Group:** One word of input AND one word of output (Input/Output pair)
- **Rack:** A memory location consisting of 8 Groups

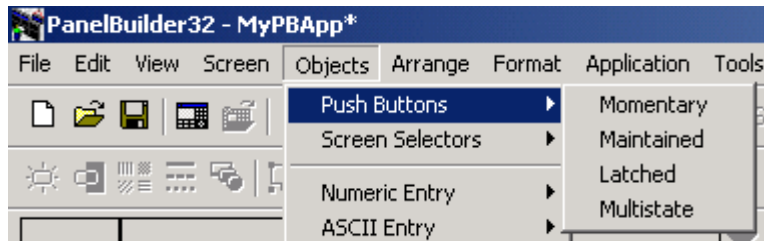
### **Conversion to SLC Addressing:**

Using the Remote I/O Configuration worksheet, you can see that each scanner module supports up to 4 Racks (0 to 3). Since each of these Racks contain 8 groups, the scanner contains 32 words of input and 32 words of output. These 32 words of input and output appear to the SLC in the slot number the scanner resides in.

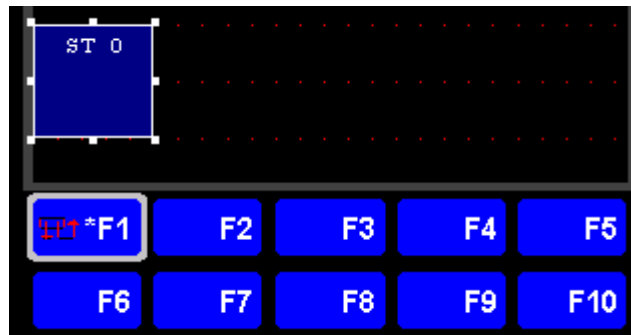
For example, if a switch is in Rack 3, Group 7, and bit 15... The address in the SLC would be I:1.31/15 in the SLC. (Refer to the Remote I/O Configuration worksheet in appendix C of the 1747-SN manual)

## *Pushbutton Objects*

- 1) With a display screen in focus (where the title bar is highlighted), you will be able to click “Objects” on the menu bar of Panelbuilder32. You will notice that you have four selections for the types of pushbuttons: Momentary, Maintained, Latched, and Multistate. Let's talk about each one of these selections:
  1. **Momentary**: Writes a value of 1 or 0 to the processor when it is pressed, then writes the opposite value when it is released.
  2. **Maintained**: Writes a value of 1 or 0 to the processor when it is pressed, then writes the opposite value the next time it is pressed.
  3. **Latched**: Writes a value of 1 or 0 each time it is pressed – Other conditions (such as ladder logic) are required to change the state of the bit.
  4. **Multistate**: Is capable of reflecting the state of multiple numeric values in the processor (other than simply 0 or 1)



- 2) For this first Example, we will add a Maintained Pushbutton. Click Objects | Push Buttons | Maintained to get the object, then let's draw the object on your screen just above the F1 key as shown.



- 3) Next, we need to get to the configuration for this button. Double click the object you just drew on your screen to get to the configuration menu. We will be writing a SINGLE BIT with the initial state of 0. Since we have a keypad terminal only, we must tie this pushbutton to a function key. Since we drew the object above F1, we will be sure F1 is selected.

The configuration menu shows the following settings:

- Type: Maintained
- Write: Single Bit (selected), Value
- Data Format: (empty dropdown)
- Initial State: 0
- Hold Time: (empty dropdown)
- Input: Function Key, F1 (selected), Touch Cell (unchecked)
- Contacts: Normally Open (selected), Normally Closed
- Data Tags: Edit Tag... button, Write Tag: (empty dropdown), Indicator Tag: (empty dropdown), Handshake Tag: (empty dropdown)

- 4) Let's talk about the **write** tag, and the **indicator** tags. The write tag will reflect the memory location we want to change when the operator presses the button. The indicator tag is the memory location we look at to reflect the state of the button. Think about a lighted pushbutton you might have in your plant. The pushbutton contacts themselves might be wired to an input on the PLC, but the light on the pushbutton may be wired to the actual output that indicates that a process has been started. That is why we have two different tags for our pushbutton objects.
- 5) Click the “Edit Tag” button so we can configure our tag to write to the PLC.

The 'Data Tags' section is shown with the 'Edit Tag...' button highlighted in red. Below the button are three empty dropdown menus for 'Write Tag', 'Indicator Tag', and 'Handshake Tag'.

- 6) Let's name this tag, "MyFirstPushButton". It will be a BIT tag with the initial value of 0. Use the Pull-Down tab to select "MyPLC" from your list of available PLC's we can associate with this tag. Let's use I:1.1/0 as the address we will be writing to. We will save word 0 for block transfers if we need them later on. Press the OK button.

- 7) For now, let's set up the indicator tag to be the same as the write tag as shown:

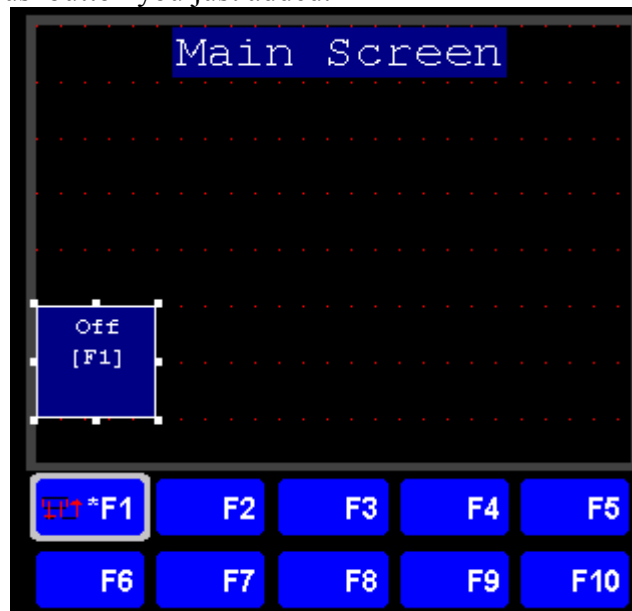
- 8) Next, let's configure the indicator states for our pushbutton. This will tell the operator when the bit is high, or when it is low. At the top of your pushbutton configuration dialog screen, click "states".

	Message Text	Graphic	Blink
0	ST 0	None	<input type="checkbox"/>
1	ST 1	None	<input type="checkbox"/>
E	Error	None	<input type="checkbox"/>

- 9) When our value is 0, we will display the text “Off [F1]” When our Value is 1, we will display the text “On [F1]. We want to be sure to include the [F1] text, so the operator knows which function key is associated with the object. This is not an issue with touch pad only Panelview terminals, because the operator just touches the object itself. Configure any other properties you wish to display on each state at this time.

	Message Text
0	Off [F1]
1	On [F1]
E	Error

- 10) When you are finished, download your project and verify the data table in the PLC corresponds to the state of the pushbutton you just added.





## Multistate Indicators

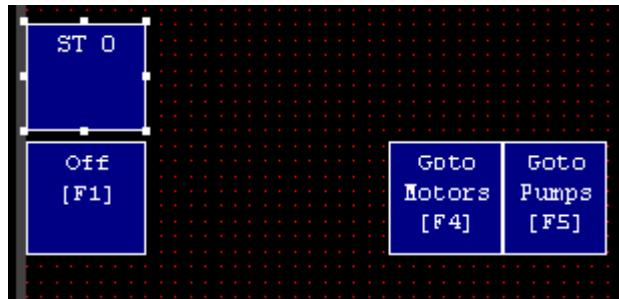
Multistate indicators look at a value in the processor, and reflect it's state based on the value of the tag it's referencing.

In this example, we will create a multistate indicator that looks at O:1.1/0. When this bit is high, we will display the text “Bit On”. When this bit is reset, let's display the text “Bit Off”.

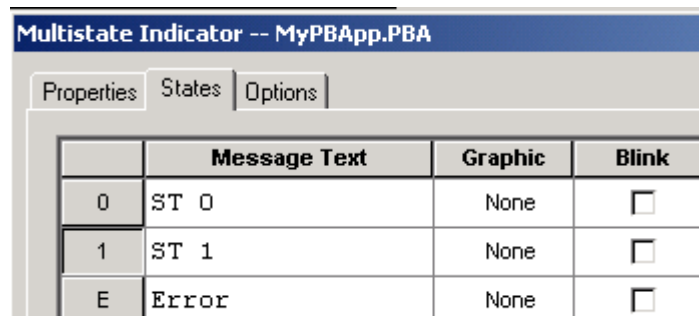
- 1) First get the Multistate indicator object. (Objects | Indicators | Multistate)



- 2) Next, draw the object on your screen. In this example, we'll just put the bit on the main screen.



- 3) Double click the multistate indicator object to access it's configuration screen.
- 4) On the STATES tab, delete all states except for 0, 1, and error as shown:



5) Configure state 0 to read “Bit Off” and State 1 to read “Bit On”

	Message Text	Graphic	Blink
0	Bit Off	None	<input type="checkbox"/>
1	Bit On	None	<input type="checkbox"/>
E	Error	None	<input type="checkbox"/>

6) On the Properties tab, configure this indicator to look at a single bit. We have to create a tag to relate this indicator to O:1.1/0, so click the button “Edit Tag”.

Multistate Indicator -- MyPBApp.PBA

Properties States Options

Read

Single Bit  
 Least Significant Bit  
 Value

Data Format

Trigger State 0 When:

Bit = 0  
 Bit = 1

Print

Read Tag:

Edit Tag...

7) Name your tag as shown, select the node you are communicating with, and set the address as O:1.1/7

Tag Form

Tag Name: MyFirstIndicator Data Type: Bit

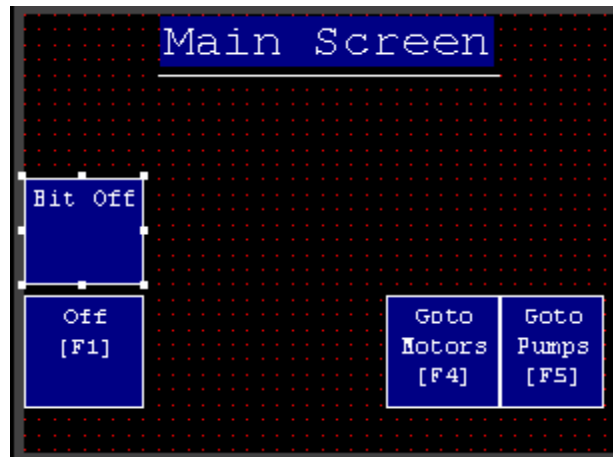
Description:

Node Name: MyPLC Tag Initial Value: 0

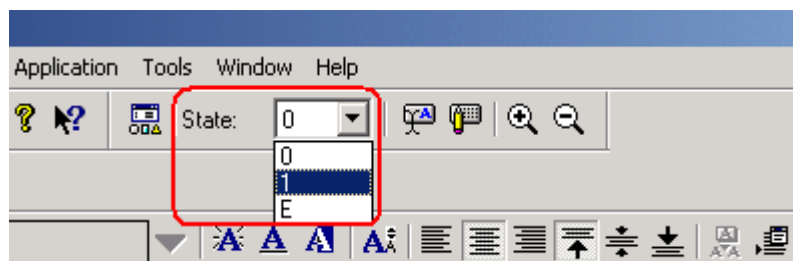
Tag Address: O:1.1/0

OK Cancel Help

8) Press OK. When finished your screen will appear as shown:



9) To see what the object will look like in various states, you can access the pull down tab in the tool bar to change the state of your object.



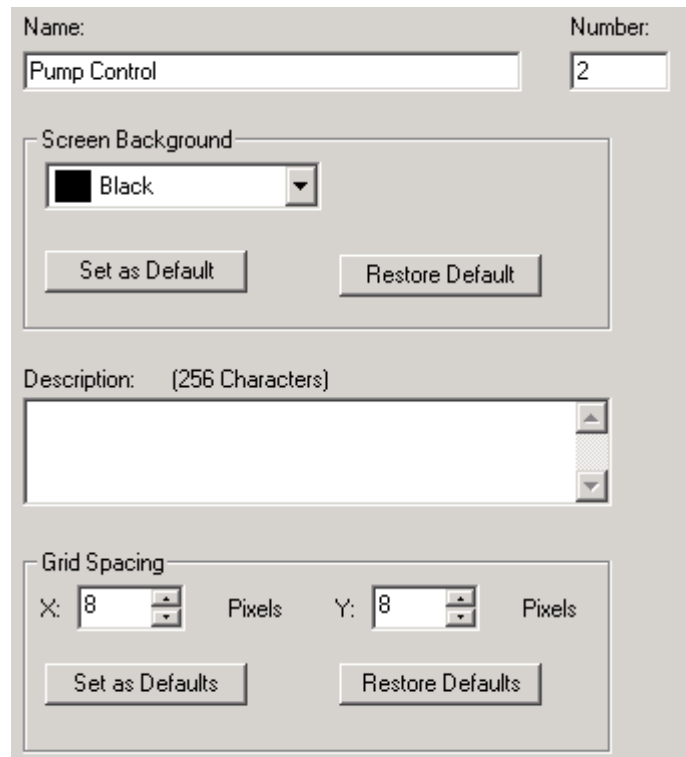
## Navigation

- 1) Now that you know how objects can be added to screens and how to configure those objects, we will set up navigation between screens. First, we will add a second screen to our project. We'll just call this screen "Pump Control".

- 2) To create a new screen, right click "Screens" in your application tree, and select "New"



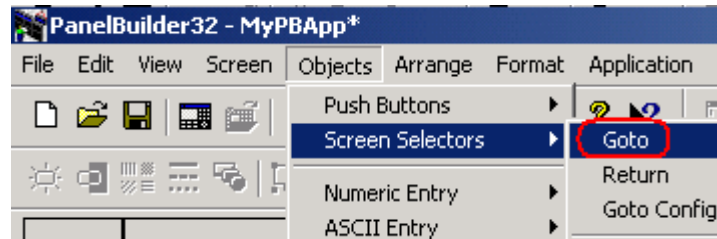
- 3) Let's name this screen "**Pump Control**". Make the grid **8 x 8** to give us more resolution on the grid for sizing objects. Press **OK** when finished.

A screenshot of a dialog box for creating a new screen. The 'Name' field contains 'Pump Control' and the 'Number' field contains '2'. The 'Screen Background' is set to 'Black'. The 'Grid Spacing' is set to '8' pixels for both X and Y. There are 'Set as Default' and 'Restore Default' buttons for both the background and grid spacing sections. A description field is empty and has a '(256 Characters)' limit.

- 4) Now, let's go back to screen 1 so we can add a navigation button allowing us to get to our pump screen.



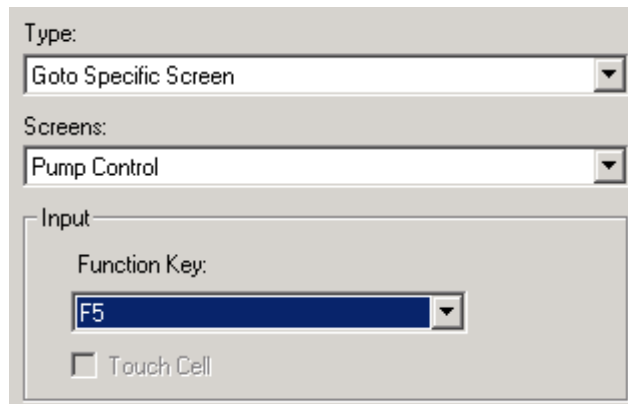
- 5) Next, be sure you have focus on screen 1, then choose objects from the menu bar, screen selectors, and choose the “Goto” object.



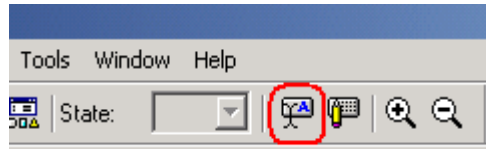
- 6) Draw the object on your screen just over the F5 key as shown:



- 7) You will notice the button automatically became associated with the F2 key by default. We will change this. Double click the object on your screen. Configure this object to go to the “Pump Control” screen when the “F5” function key is pressed.



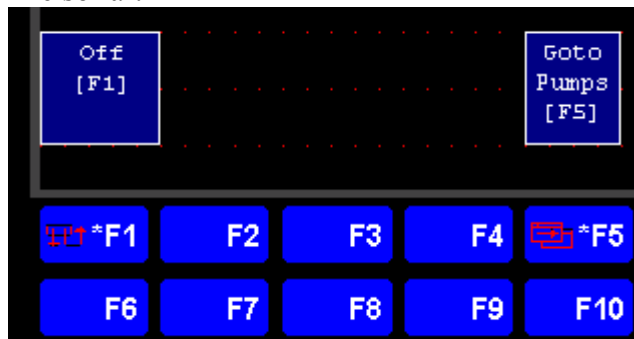
- 8) Next we need to change the text on the button. Click the “Inner Text” button as shown to edit the text:



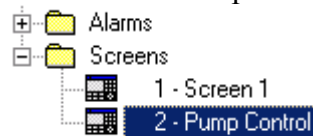
- 9) Change your text as follows: (Note: /\*R\*/ = Carriage return)



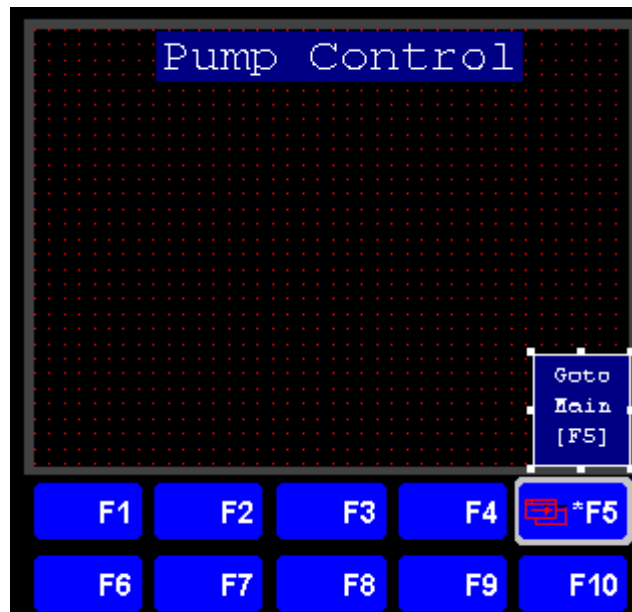
- 10) Now that we have added our button to the Main Screen, we need to add a button to the Pumps Screen so we can get back to the main screen... Otherwise once the operator navigates to the Pump screen, he will be stuck there and have to reboot the Panelview Terminal to get back to the main screen. This is a common mistake among Panelview Programmers. Here is what our Main Screen looks like so far:



- 11) Lets add our button to get back.... Go to the Pump Control Screen.



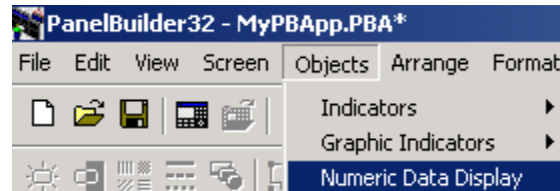
- 12) Add a text object at the top of your screen that indicates to the operator which screen he is on. Add a Goto button to this screen to get back to the Main Screen as shown. Once you draw the object above the F5 key, be sure to double click the object for configuration.



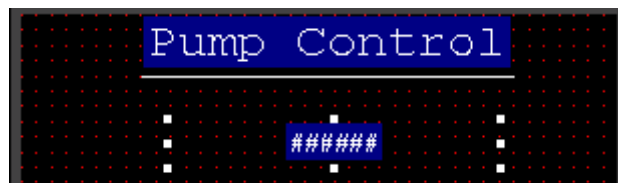
## *Numeric Data Display*

The Numeric Data Display object allows you to display a numeric value from a memory location in the processor. Since the PLC is controlling this display on the Panelview terminal, this would be considered an output from the PLC.

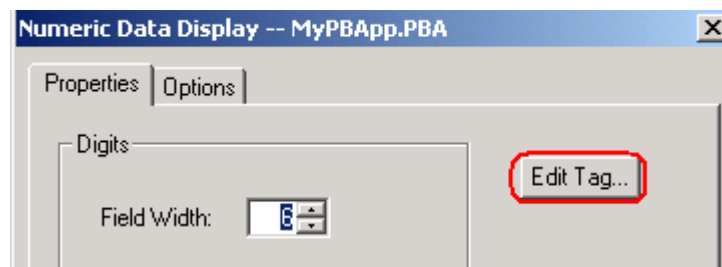
- 1) Let's put this on the Pump Control screen, so with this screen in focus, click Objects | Numeric Data Display.



- 2) Next, draw the Numeric data display object onto your Pump Control screen.



- 3) Double click the object to get to it's configuration, then choose "Edit Tag". We need to associate this numeric data display object with a memory location in the PLC.





- 4) Set up your tag as follows: Notice that you will be writing to an integer this time instead of a bit address, so our address needs to be specified to the word level!

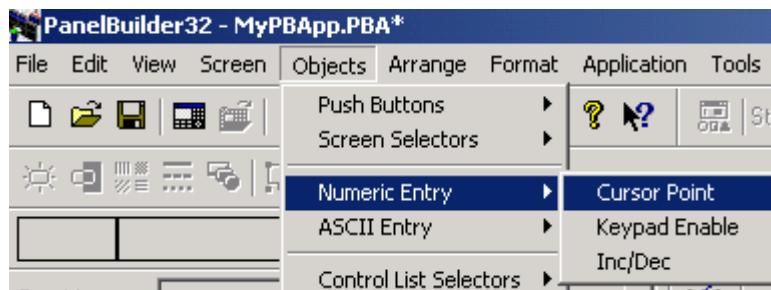
The screenshot shows the 'Tag Form' dialog box with the following configuration:

- Tag Name: MyNumericDisplay
- Data Type: Signed Integer / INT
- Description: (empty)
- Node Name: MyPLC
- Tag Initial Value: 0
- Tag Address: 0:1.2
- Scaling: Scale = 1, Offset = 0
- Data Entry Limits: Min = -32768, Max = 32767

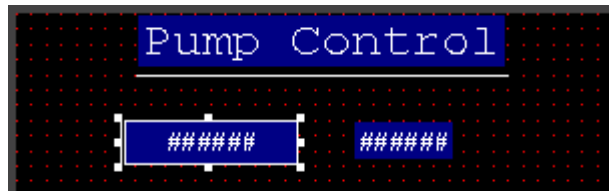
# Numeric Entry

The Numeric Entry object allows the operator to enter a numeric value onto the Panelview terminal. This value is then sent to a memory location in the PLC, so this would be considered an INPUT from the PLC's point of view. There are two types of numeric entry: Keypad Enable, and Cursor Point. Keypad Enable associates the object with a function key. When the function key is pressed, a scratchpad will open up allowing the operator to enter a numeric value. Cursor point allows us to use the arrow keys to select the object we want to enter data into. For this example, we will use the cursor point object to enter data.

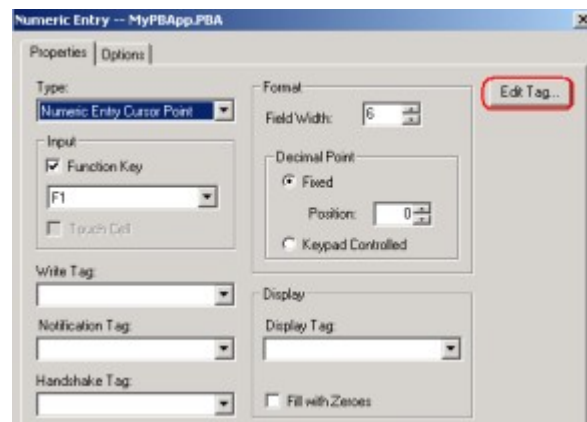
- 1) We will place this numeric entry object on the Pump Control screen, so with this screen in focus click Objects | Numeric Entry | Cursor Point.



- 2) Next, Draw the object on your Pump Control screen as shown:



- 3) Double click the object to get to it's configuration, then choose "Edit Tag". We need to associate this numeric entry object with a memory location in the PLC.



- 4) Next, fill out the appropriate fields in the tag editor. Since this is going to the PLC, we will consider this to be an input.

The screenshot shows the 'Tag Form' dialog box. The 'Tag Name' field is set to 'MyNumericEntry', the 'Data Type' is 'Signed Integer / INT', the 'Node Name' is 'MyPLC', and the 'Tag Address' is 'I:1.2'. The 'Tag Initial Value' is set to '0'. The 'Description' field is empty. The 'OK', 'Cancel', and 'Help' buttons are visible on the right side of the dialog.

- 5) Set up your display tag to reflect the memory location we are writing a value to for this example:

The screenshot shows the 'Display' dialog box. The 'Display Tag' dropdown menu is set to 'MyNumericEntry'. The 'Fill with Zeroes' checkbox is unchecked.

# Alarms

Alarms are designed to alert the operator if certain conditions are present such as Low temperature, Low Oil Levels, and various machine failures. The Panelview terminals have a dedicated alarm screen for this purpose. Be sure you are not relying on your Panelview as a primary warning device that could cause damage to personnel, equipment, or cause significant downtime.

## Alarm Banner

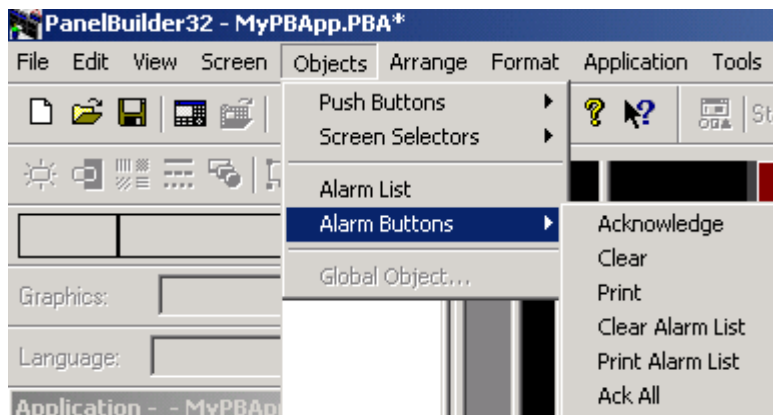
- 1) The alarm banner appears over other screens when an alarm is triggered. To create an alarm banner, right click “Screens” in the application tree.



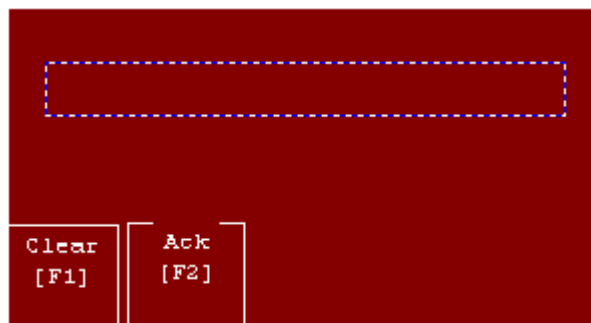
- 2) Once you create your alarm banner screen, go to Screen | Properties and increase the resolution of your screen to 8 x 8 as we did for the pumps screen. This will allow us more precise positioning of objects we place on the alarm banner screen.



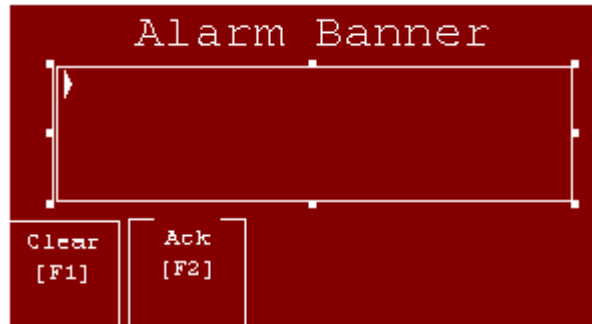
- 3) Next, Let's Look at the available buttons you can add to this alarm banner. Click Objects | Alarm Buttons from the menu bar. Let's discuss each of these buttons individually:
  1. **Acknowledge:** Marks the alarm as acknowledged that is currently displayed in the alarm banner. The alarm banner will disappear, but this does not clear the alarm condition.
  2. **Clear:** This button makes the banner disappear from your screen. This does not acknowledge the alarm, nor does it clear the alarming condition. You will need to make sure this button exists on the alarm banner to clear the banner if no conditions can be acknowledged.
  3. **Print:** This button will print a single alarm if you have an RS232 printer. (Note: The RS232 port cannot be used for downloading an application if it is configured for printing)
  4. **Clear Alarm List:** Clears the Alarm List and the Alarm Banner even if the alarm condition still exists.
  5. **Print Alarm List:** Prints All the Alarms in the list.
  6. **Ack All:** This button allows the operator to acknowledge all alarms that have not yet been acknowledged. This will cause the alarm banner to disappear from the operator screen, but does not clear the condition that caused the alarm.



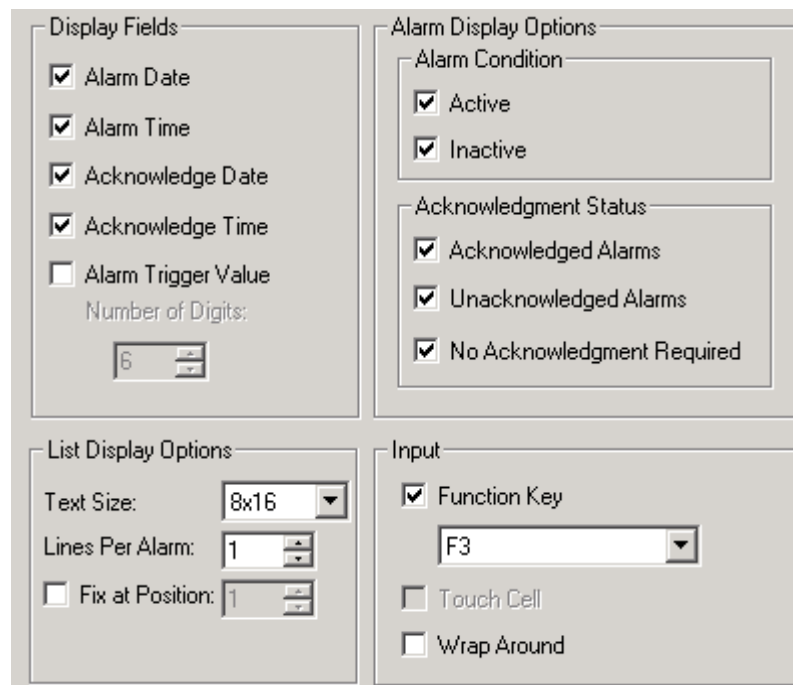
- 4) Let's move the Clear button over the F1 Button, and add an Acknowledge button to the Alarm banner. We'll place the Acknowledge button over the F2 Button. Modify the inner text of each button to reflect the function key it is associated with as shown:



- 5) Let's also add the alarm list object to your screen as shown (Objects | Alarm List), and edit the text object's inner text to read "Alarm Banner".



- 6) You can double click the alarm list to change various properties. For this example, we will just leave them at default.



- 7) Now, let's see what options we have available for alarming: In your application tree, open the alarm folder, and go into "Setup".



- 8) In the top portion of our setup, we have various options for the banner and alarm list:

A screenshot of a configuration window for alarms. It is divided into four sections: "Banner Pop Up" with radio buttons for "Covered Objects Disabled" and "All Objects Disabled" (selected); "Print Items" with checkboxes for "Alarm Date", "Alarm Time", "Acknowledge Date", "Acknowledge Time" (all checked), and "Alarm Trigger" (unchecked), plus a "Number of Digits" spinner set to 6; "Alarm Lists" with a checkbox for "Clear Lists On Power Up" (unchecked) and a "Size" spinner set to 25; and "Time Values" with "Ack. Hold Time" set to 500 ms and "Snapshot Timeout (seconds)" set to 2.

- 9) At the bottom, you have various options to configure tags that will allow the PLC to interact with the alarm list:

A screenshot of a "Remote Tags" configuration window. It contains several dropdown menus: "Ack. All Tag:", "Ack. All Handshake Tag:", "Clear All Alarm Tag:", "Clear All Alarm Handshake Tag:", "Data Tag:", "Handshake Tag:", and "Notification Tag:". Each dropdown menu is currently empty.

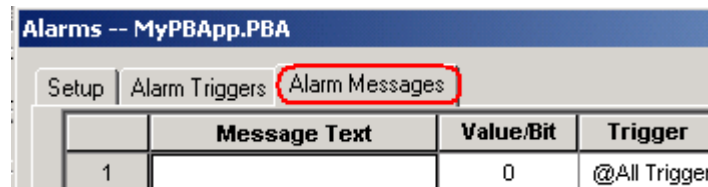
10) On the next tab (Alarm Triggers), the user can configure a list of tags which have the ability to trigger an alarm. These alarms can be triggered by the Least Significant bit (LSBIT) to Most significant bit in a priority sequence, a Bit with in a word (BIT), or when a tag reaches a certain Value. Most applications will use only one alarm trigger tag. The value of this tag (or the bits that are set within this tag) will define which message is displayed from the alarm messages tab.

1. **Ack Tag:** This is the tag the Panelview writes to the PLC to inform the PLC the tag has been acknowledged
2. **Handshake Tag:** The Panelview will toggle a bit in the PLC indicating that it has received the alarm trigger.
3. **Remote Ack Tag:** The Panelview toggles a tag in the PLC indicating that it had received a Remote Acknowledge for an alarm trigger.
4. **Remote Ack Handshake:** Toggles every time the remote Ack tag changes
5. **Acknowledge All Value:** This is the value written to the remote Ack tag when all alarms have been acknowledged for a trigger



11) Next, let's look at "Alarm Messages" This is where we configure the alarm messages based on a trigger.

1. **Text:** The message displayed when the alarm is triggered
2. **Value:** The value or bit of the trigger tag that will cause a particular message to be displayed.
3. **Trigger:** These are pre-defined on the alarm trigger tab.
4. **Ack:** Specifies whether or not the alarm text on the alarm banner must be acknowledged
5. **Print:** If you have an RS232 printer, this box will specify if the alarm condition is to be printed.
6. **Display:** Specifies whether or not the alarm text is to be displayed in the banner once the alarm has been triggered
7. **Background and Foreground:** Define colors for the alarm text
8. **Text ID:** A unique identifier that Panelbuilder32 associates with this text in the project.

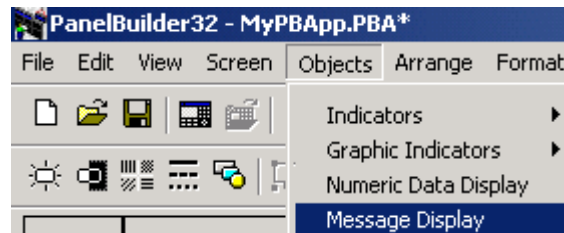




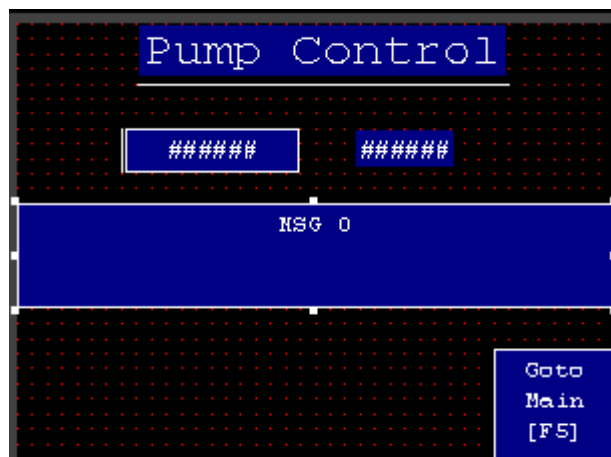
## *Message Object*

The Message Object can be used to display a particular message on a Panelview screen based on the value of a memory location in the PLC. An example might be a equipment that runs through a various sequence of events. The message display can indicate where the system is in the sequence. For this example, we will add

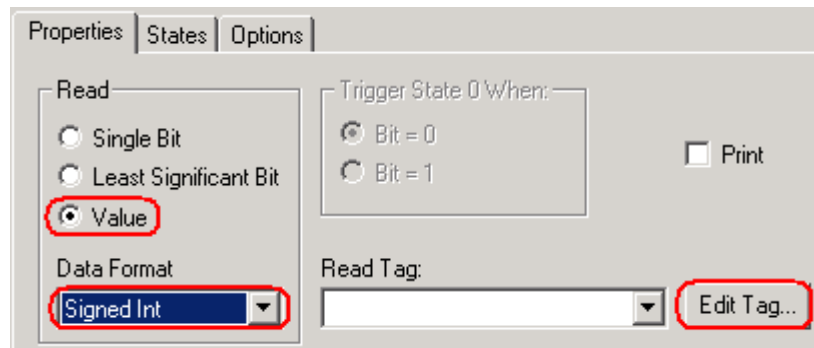
- 1) Since we will be adding this object to the Pump Control screen, be sure this screen is in focus, then click Objects | Message display from the Menu Bar.



- 2) Draw the object on your Pump Control screen as shown:



- 3) Double click the object to access the configuration screen. For this example, we will look at a Signed integer value. Choose “Edit Tag” so we can relate this message display object with a memory location in the PLC.



4) Configure your tag as shown, the press OK.

The screenshot shows the 'Tag Form' dialog box with the following configuration:

- Tag Name: MyMessageDisplay
- Data Type: Signed Integer / INT
- Description: (Empty text area)
- Node Name: MyPLC
- Tag Initial Value: 0
- Tag Address: O:1.3
- Scaling: Scale = 1, Offset = 0
- Data Entry Limits: Min = -32768, Max = 32767

5) Next we need to Configure the various states for the message display, based on the value the PLC loads into O:1.3. Configure the various states as shown below: (Note: To add additional states, right click on the states screen and choose “Insert State”.

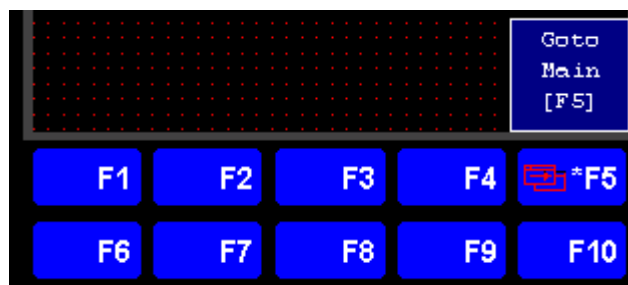
Properties States Options					
	Message Text	Value	Graphic	Blink	Fill
0	Shutdown	0	None	<input type="checkbox"/>	
1	Initializing	1	None	<input type="checkbox"/>	
2	Loading	2	None	<input type="checkbox"/>	
3	Processing	3	None	<input type="checkbox"/>	
4	Finalizing	4	None	<input type="checkbox"/>	
5	Unloading	5	None	<input type="checkbox"/>	
E	Error		None	<input type="checkbox"/>	

## Global Objects

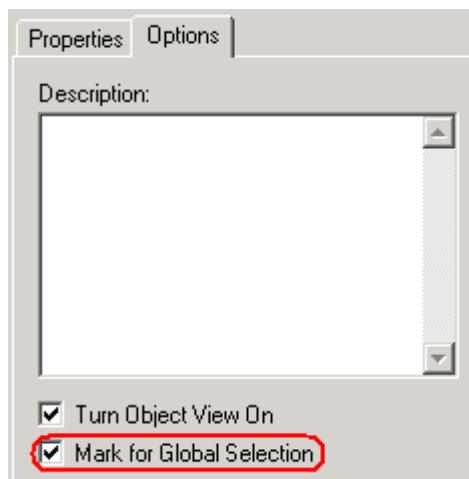
Global objects can save memory in the Panelview terminal, and development time for your application by using the same object on multiple screens.

For this example, we are going to mark our “Goto Main [F5]” Button for global selection. This will allow us to place the object on any new screen we create.

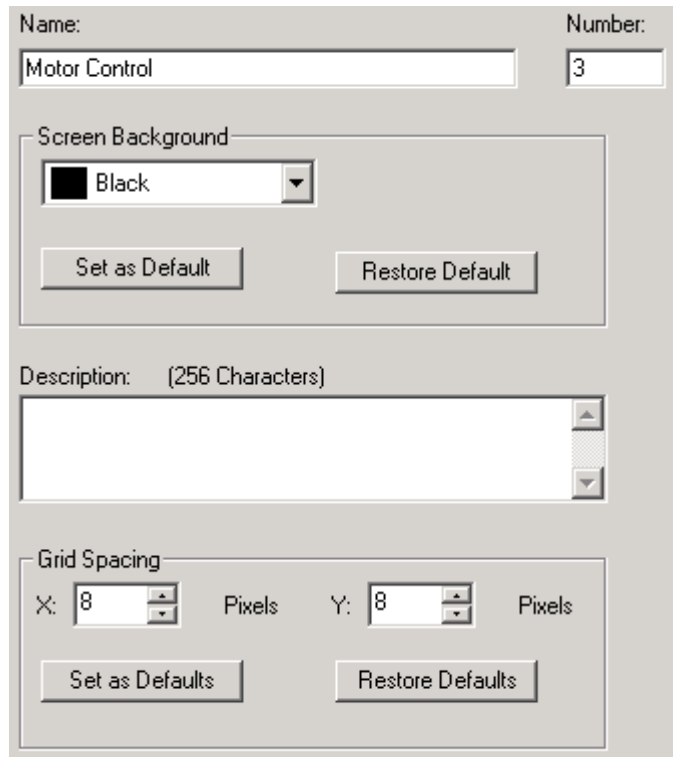
- 1) Double click the object on the Pump Screen. This will take us to the configuration for the object.



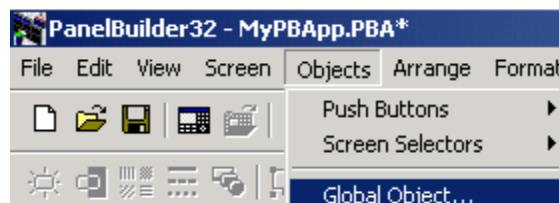
- 2) On the “Options” tab, choose “Mark for Global Selection”



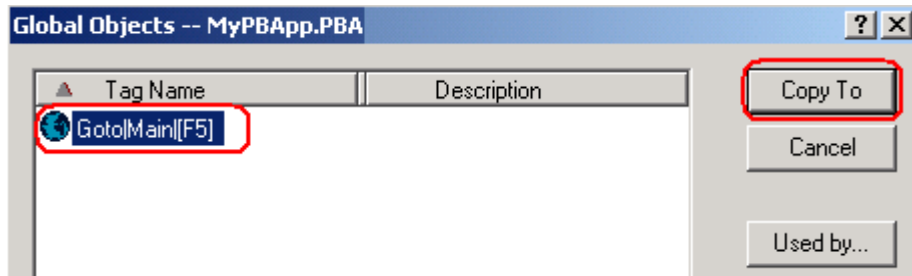
- 3) Now let's create a new screen called “Motor Control” (Right click “Screens” in the application tree to create the new screen) Let's make the grid on this screen 8 x 8 as well.



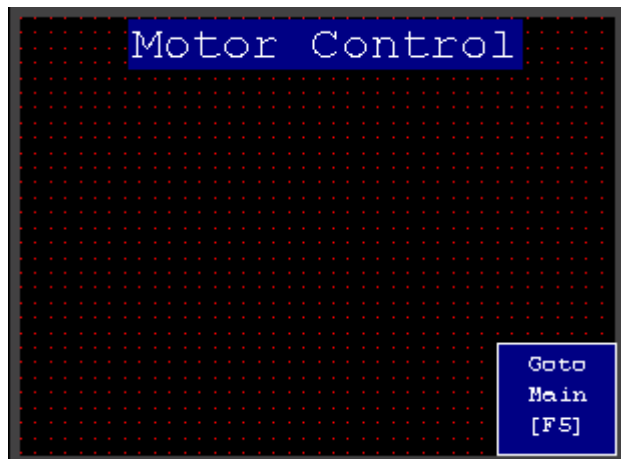
- 4) Next we will access the list of global objects, and we will select the button that gets us back to the main screen from our motor control screen. To access the list of global objects, click Objects | Global Object.



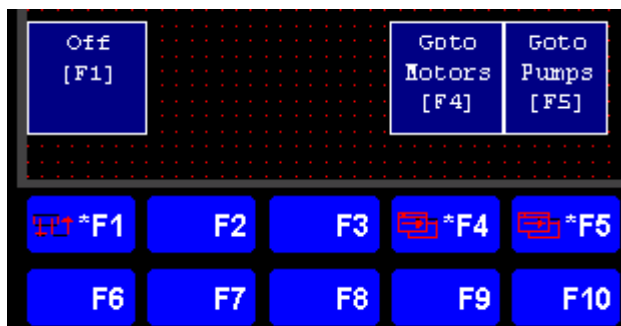
- 5) Highlight the TagName of the object you want to use (The only one currently in our list), then choose “Copy To”.



- 6) Next, Draw the object on your Motor Control Screen. Don't forget to add a text object to the top to identify the purpose of this screen.



- 7) Be sure to add a button to the main screen as well that will allow the operator to access this Motor Control Screen. We'll just use the F4 key for that. You may want to edit the properties of the main screen to allow 8 x 8 resolution as you have for the other screens.

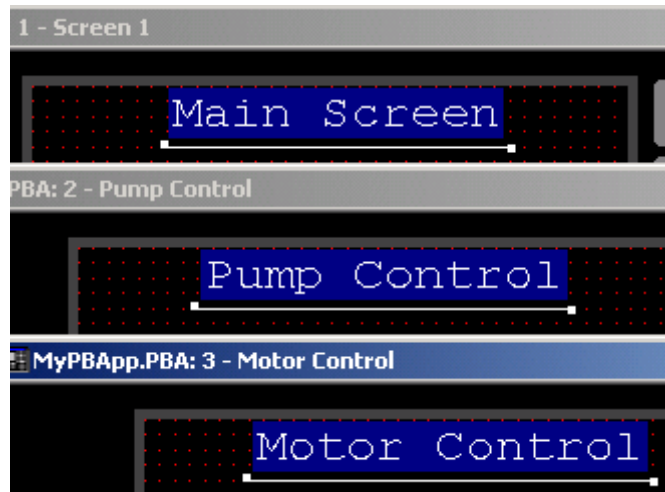


# Graphics

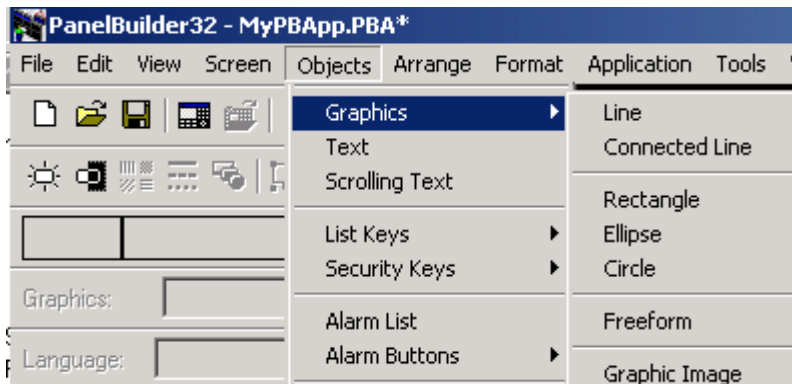
Graphics can be used to enhance the appearance of your Panelview screens. Standard Panelview terminals have some limited graphic support. Let's take a look at a few of the features you will be able to utilize in your projects:

## Simple Graphics

- 1) Click Objects | Graphics | Line. Draw a Line under the Title of each page. This is probably the simplest graphic object.



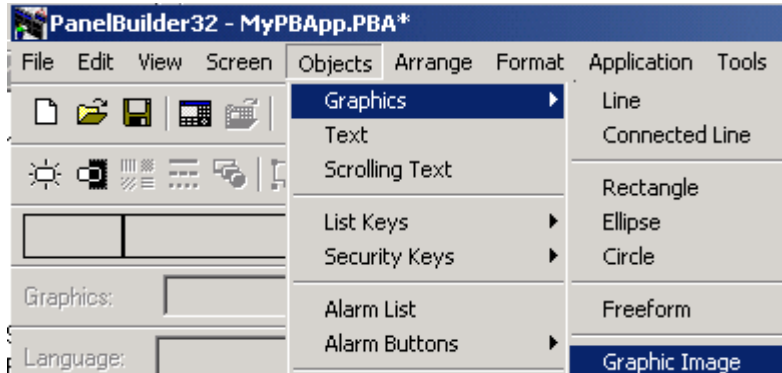
- 2) You will also notice that you can draw connected lines, rectangles, ellipses, circles, and freehand lines.



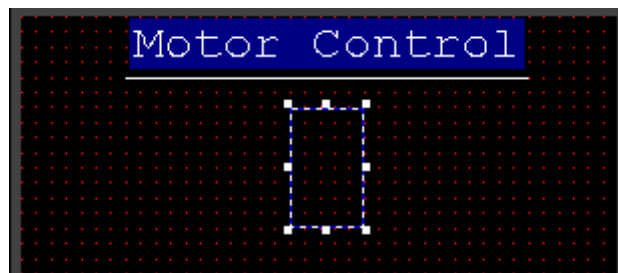
- 3) The object we'll spend a little more time with for this exercise will be the graphic image object. Some graphics images are already defined in Panelbuilder32 software, while others that we use in the project can be imported from an external source.

## ISA Images

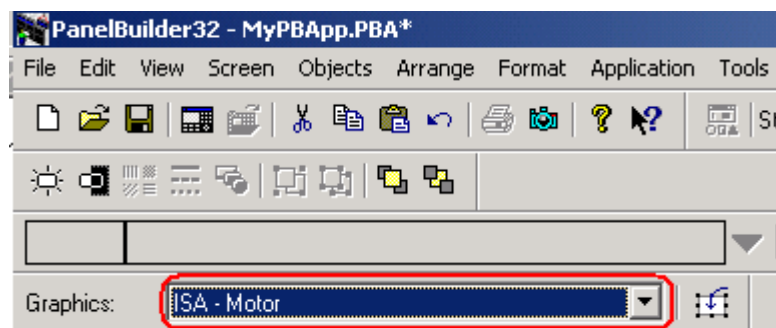
- 4) Click Objects | Graphics | Graphic Image.



- 5) Now that we have the Graphic Image object, Let's draw the object onto our Motor Control screen .

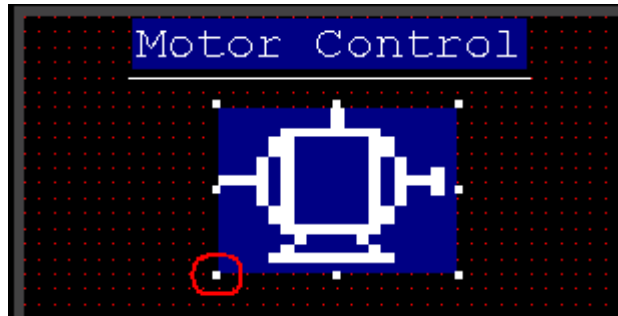


- 6) Select the pull down tab in the graphics tool bar, and select the ISA Image of a motor.



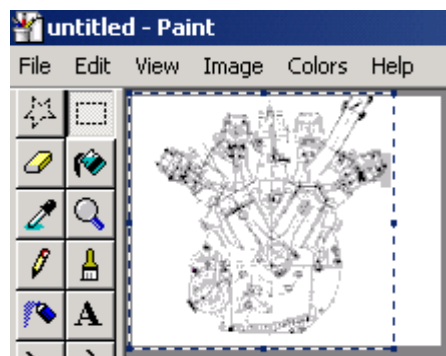


7) You can use the handles on the image to size the image according to your application.

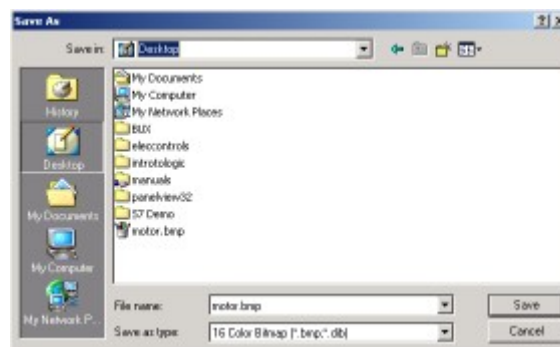


## Imported Images

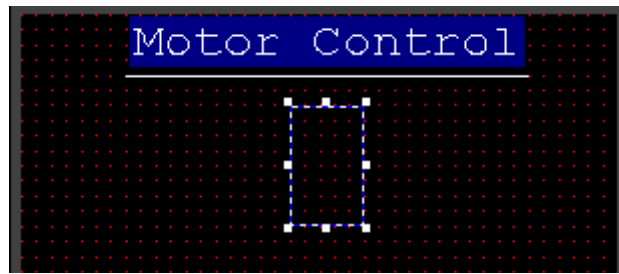
8) In this example, we have a picture of a motor in Microsoft Paint that we are going to import onto the Motor Control Screen.



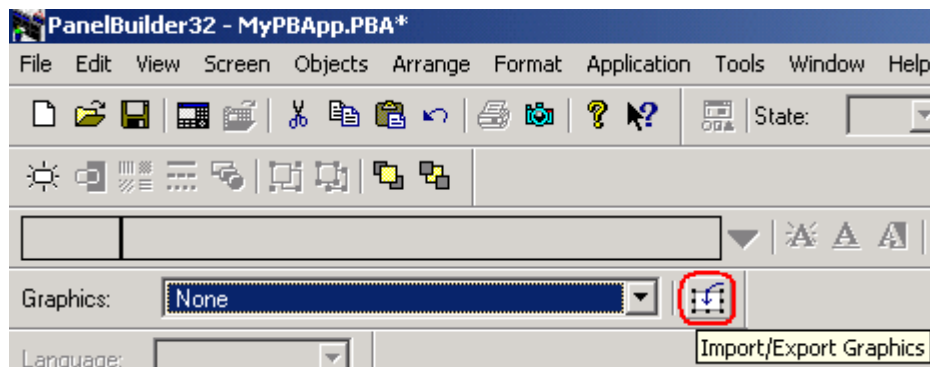
9) First, in paint, we are going to save this as a 16 color bitmap. Click File | Save As, we we will put this file called motor.bmp onto the desktop.



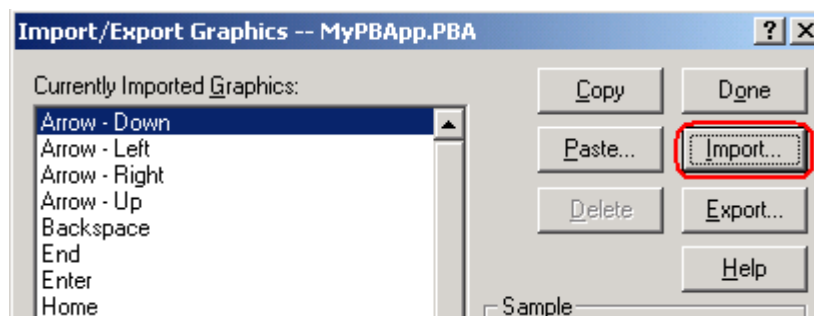
10) Next, click Objects | Graphics | Graphic Image, and draw the object onto your motor control screen.



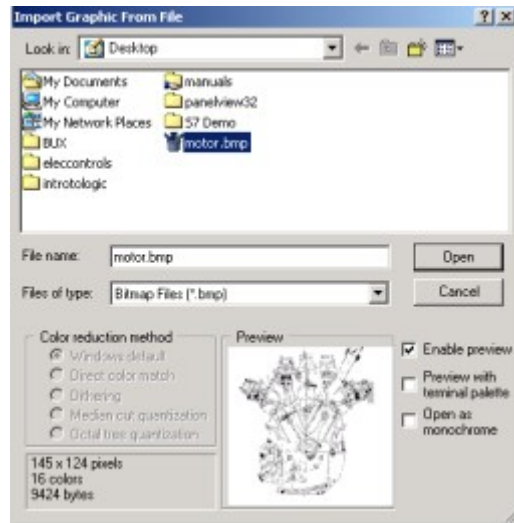
11) Click the Import | Export Graphics Tool in the Graphics tool bar.



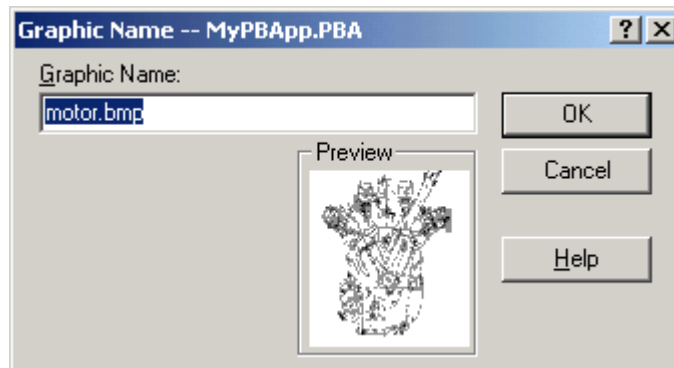
12) Click the "Import" button within the utility.



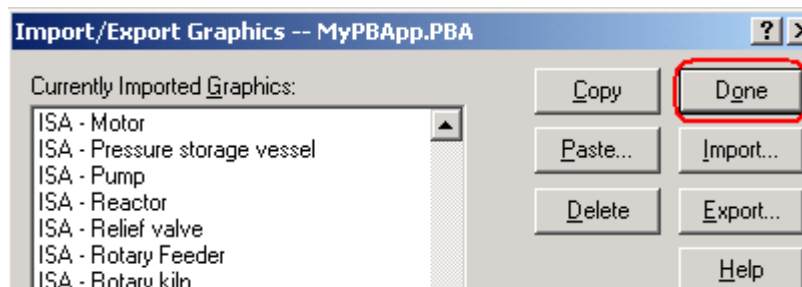
13) Next, Browse for the file on your desktop.



14) Click OK, then you will be asked to confirm the graphic image, then press OK again.



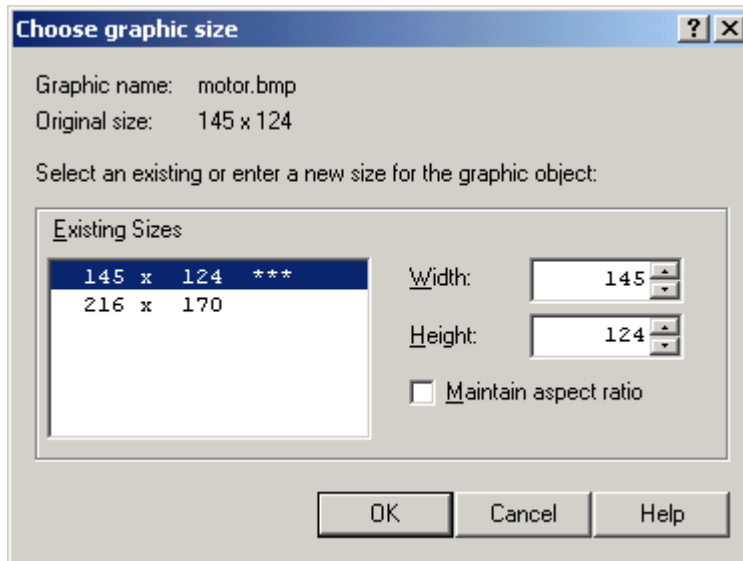
15) Next click DONE. Your graphic image will now be imported, but we aren't finished yet.



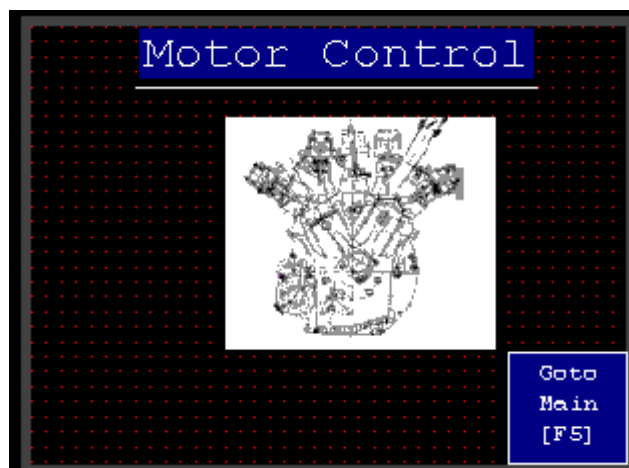
16) Next select motors.bmp from the list of available graphics in the graphic tool bar.



17) Verify the graphic size, then press OK.



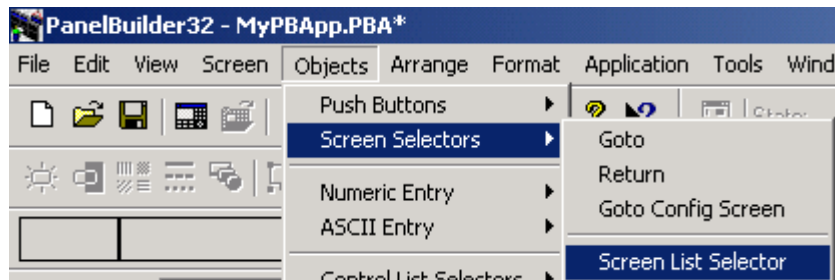
18) Your graphic should now be on the screen.



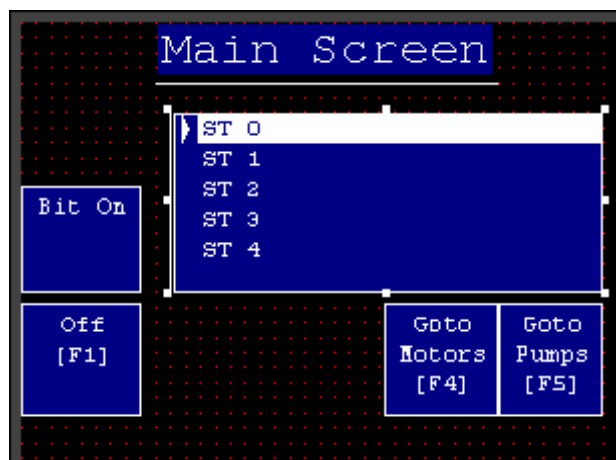
## Screen List Selector

The screen list selector allows the operator to navigate to a specific screen through a list rather than through various “Goto” buttons placed on the screen.

- 1) For this example, we will add a screen list selector to the Main Screen. Click Objects | Screen Selector | Screen List Selector from the menu bar.

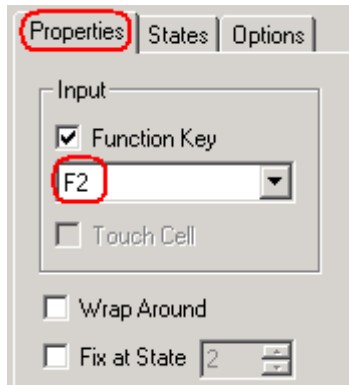


- 2) Draw the screen list selector object onto the Main Screen.



- 3) Next, Double click the Screen List Selector object to access the configuration screens

- 4) On the Properties tab, we will associate this screen list selector object with the F2 Function Key.



- 5) Next. Let's go to the states tab and configure the available screens the operator will have access to. Screen 2 will be for Pumps, and Screen 3 will be for Motors.

	Screen	Message Text
0	2 - Pump Control	Pumps
1	3 - Motor Control	Motors

- 6) Next you can download and test your new screen selector. When finished your screen should look similar to the image below: (You may want to label your selector so the operator knows what function key to associate with the selector)

