1010-1001A Rev 01

WEINTEK

Human Machine Interface

WEINTEK

MT500

Touch Screen for Industrial Applications

http://www.weintek.com

Table of Contents

EZware Support 1
About Your Documentation 1
What You Need 2
HMI Basics 2
What is a MT5xx HMI? 4
List of Features 6

Chapter 1 - Installation of HMIs 9
Before You Begin 9
Control Panel Design Guidelines 10
Control Panel Grounding 11
Connect HMI Chassis Ground to Control Panel 11
Power Supply Selection 12
Cable Routing and Noise Immunity . 12
Installation 13
Connect the HMI to Power 13
Connect the HMI to the PLC 15
Panel Preparation 17
Mount the HMI to the Panel 17
Configuration Wiring 19
Connect the HMI to the PC for Configuration 19
Factory Configuration 21

Chapter 2 - Creating Your First Project 23
Before You Begin 23
Connecting HMI to Computer 23
Starting EZware-500 24
Creating a Sample Project 25
Setting the System Parameters 26
Creating a Startup Window 26
Creating a Popup Window 28
Finishing Up 33

Chapter 3 - Simulator Mode 37
The Simulation Screen 37
Troubleshooting Tools During Simulation 39
Capturing Simulation Screens to Use as Documentation 39
Wiring for Normal Simulation Mode 40

Chapter 4 - Using EZware-500 43
Overview 43
The EasyManager 43
Communications Settings 44
Operating Modes 44
The EasyASCIIFontMaker 45
EasyBuilder's Default Text Fonts 45
Using the EasyASCIIFontMaker 45
The EasyBuilder Application 47
Printing Projects 50
Editing and Creating Screen Objects 50
General Settings 69
Restart the HMI Automatically after a Project is Downloaded 81
Automatically Save and Compile the Project 81
Compact Flash 82
Connecting Multiple HMIs to one PLC . . . 82
Hardware Connection 82
Software Connection 83
Sharing Data Between the Master HMI and Slave HMI 83
Connecting Multiple HMIs Using the Ethernet Port 84
Sharing Data between the Master HMI, Slave HMI and PLC 88
Networking Multiple Controllers . . . 88
Using the Aux Port 90

Chapter 5 - Creating Windows 95
Windows Fundamentals 95
Opening a Window 95
Creating a New Window 96
Setting the Window Security Level 107
Assigning Underlay Windows 108
How to Display Underlay Windows . . 108
Rules That Apply to Underlay Windows 109
Deleting a Window 110
Using Base Windows 110
How to Display Base Windows 111
Using the Common Window 122
Optional settings 124
Changing the Active Common Window 128
Using the Fast Selection Window 130
Using the Fast Selection Key 133
Changing screens using the Fast Selection window 133
Changing the Fast Selection Window 136
Using the Task Bar 140
Using the Window Bar 140
General Settings 142
Creating a Message Board 143
Optional Parameters 143

Chapter 6 - Creating Graphic Objects 147
Drawing Objects 147
Using the Drawing Tools 147
Using Text 150
Predefined Shapes and Bitmaps 151
Using a Predefined Shape 152
Using a Predefined Bitmap 154
Graphics Libraries 155
What are 'states'? 156
Using Shape Libraries 156
Using Bitmap Libraries 165
Using Group Libraries 171

Chapter 7 - Creating and Using Databases and Languages 177
Creating and Using the Tag Library 177

Importing and Exporting the Tag Library 178
Using the Tag Library 179
Creating the Label Library 180
Importing and Exporting the Label Library 183
Using the Label Library. 183
Using Languages with the Label Library 185

Chapter 8 - Representing Data with Graphics Objects. 187
Using Internal Data Memory of HMI 187
Non-volatile Storage of System Parameters 193
PLC Settings. 193
General Settings 193
Security Settings. 194
Representing PLC Coil Registers 194
The Bit Lamp Object. 194
The Set Bit Object 198
Toggle Switch Object 199
Representing PLC Data Registers 201
The Word Lamp Object 201
The Set Word Object 205
The MultiState Switch Object . . . 207
The Numeric Data Object. 209
The Numeric Input Extend Object 213
The ASCII Data Object 218
The Moving Shape Object 223
The Animation Object 226

Chapter 9 - Using and Creating Keypads 231
How to Create a Keypad 231
Displaying and Using a Keypad 235
Using the Built-In Numeric Keypad in EasyBuilder 239

Chapter 10 - Bar Graphs, Meters, and Trends . 247
Creating Bar Graphs 247
Creating Display Meters 251
Creating Trend Displays 254
Creating XY Plots 256

Chapter 11 - Capturing Alarms and Events . . . 261
Using Alarms 261
Monitoring Alarms with the Alarm Scan Object. 261
Displaying Alarms using the Alarm Display Object 262
Displaying Alarms using the Alarm Bar Object 268
Using Events 270
Monitoring Events With the Event Log Object 270
Displaying Events Using the Event Display Object 272
Using Events for an Alarm History 275
Using the Alarm History 279

Chapter 12 - Data and Recipe Transfer Objects 281
Using the Data Transfer Object 281
Using the Recipe Transfer Object 282
Creating a Recipe. 286

Chapter 13 - Macros 287
Using Macros 287
What's new with macros in Version 2.6.2? 287
What's New in Version 2.6.0 . . . 287
What's New in Version 2.5.2 . . . 288
Macro Sample and Implementation 288
Variables, Declarations and Memory Usage 291
Memory Usage: 291
Variable Declarations. 291
Variable Initialization 292
Array initialization 292
Reserved Words 293
Operator 293
Order of Precedence 293
Main Functions and Sub-functions. 294
Local and Global Variables 294
Creating Variable Arrays 294
Using Macros Within Recipes . . . 295
Using 32-bit Registers Within Macros 296
Using Floating Point Registers Within Macros. 296
Statements, Conditions & Expressions 296
Function calls and passing parameters 299
Reading & Writing External Registers in a Macro 299
Precautions, tips & tricks when using Macros 300
Compiler Errors & Error Codes 301

Chapter 14 - Using a Printer With the MT508/550 305
Supported Printer Models 305
Non-supported Printer Models . . 306
Configuring the MT508/550 for a Printer 306
Printing a Window 306
Using a Function Key 306
Using Simulation Mode 307
Printing a Report 308

Appendix A - Specifications 309
MT506 309
MT508 310
MT510 311

Appendix B - Dimensional Outlines 313
MT506 313
MT508 314

MT510 315
Appendix C - Panel Cutout Dimensions 317
MT506 317
MT508 317
MT510 318
Appendix D - Libraries 319

Introduction - Welcome

Welcome to the Weintek' MT5xx of Operator Interface Terminals (HMIs). Using graphic HMIs has never been easier. This powerful family of graphics operator interface terminals connects to programmable logic controllers (PLCs) to provide the human-machine interface in industrial applications. The MT5xx has several features not found in other graphic HMIs. This manual explains the operation of the MT5xx HMIs and how to implement the many available features using the EZware-500 Configuration Software.

EZware Support

HMI Models Supported

For the latest list of MT5xx Models supported by EZware, please visit our website at www.weintek.com

PLCs Supported

For the latest list of PLCs and controllers supported by the MT5xx HMIs, please visit our website at www.weintek.com

About Your Documentation

Weintek provides many resources to allow you to get the most out of your MT5xx HMI.

- *MT5xx Operation Manual* (shipped with EZware-500 as a PDF file) - describes installation, general operation and features of the MT5xx using EZware-500 configuration software.
- *Controller Information Sheets* - important information specific to each supported protocol.
- *EZware-500 On-line Help* - covers the operation of EZware. Always available by clicking **Help Topics** from the **Help** menu in EasyBuilder.

Conventions

When using EZware-500, there are usually several ways to perform a task. For example, if you want to copy a graphics object, you can:

- Click the Copy command on the Edit menu.
- Click the Copy button on the Standard toolbar.
- Press the CTRL + C keys on your computer.

In most cases, we will describe each method when the task is first discussed. The menu method is then used whenever the task is used in later procedures. Other conventions used in this book are listed in the following table.

Convention	Meaning
Bold	Characters that you must type exactly as they appear. For example, if you are directed to type **a:\setup**, you should type all the bold characters exactly as they are printed.
Italic	Placeholders for information you must provide. For example, if you are directed to type *filename*, you should type the actual name for a file instead of the word shown in italic type. Italics are also used to indicate a glossary term.
ALL CAPITALS	Directory names, file names, key names, and acronyms
KEY1+KEY2	A plus sign (+) between key names means to press and hold down the first key while you press the second key.
click	Refers to clicking the primary mouse button (usually the left mouse button) once.
Double-click	Refers to quickly clicking the primary mouse button (usually the left mouse button) twice.
Right-click	Refers to clicking the secondary mouse button (usually the right mouse button) once. Right-clicking usually opens shortcut menus.

The following table identifies symbols and margin icons.

Icon	Meaning
▶	Identifies a procedure.
	Indicates a reference to additional information.
	Indicates an important note.

What You Need

The following items are needed to configure and operate your HMI.

Configuration Software	EZware-500
Configuration Cable (HMI to PC) Y-adapter cable[1]	7431-0098
Personal Computer[2]	User Provided
Power Cable	6030-0009
24VDC Power Supply	User Provided (for details refer to Appendix A: Specifications)
PLC	User Provided
Controller Information Sheet	Weintek provides Controller Information Sheets which contain important information specific to each PLC. **Please locate the sheet that corresponds to your PLC on our website.**
Communication Cable (HMI to PLC)	Refer to Weintek' Tech Note 1061 for a list of available cables or build your own using the cable diagrams available on our web site (www. maple-systems.com).

[1]Allows you to connect HMI to PC using RS232 and a PLC that uses RS485 at the same time

[2]Computer requirements include at least a Pentium 90Mhz PC, 16MB RAM, 10MB available hard disk space, VGA video controller, Microsoft Windows 95, 98, 2000, or NT, and one available RS-232 serial port.

HMI Basics

Operator Interface Terminals (HMIs) provide much more versatility than traditional mechanical control panels. An HMI allows a plant floor operator to monitor current conditions of a control system and, if necessary, to initiate a change in the operation of the system. HMIs connect to programmable logic controllers (PLCs) typically through the PLC's serial communications port. The HMI can be programmed to monitor and/or change current values stored in the data memory of the PLC.

HMIs can have either text-based or graphics-based displays. A text-based HMI can display printable text characters but no graphics. Some text-based HMIs can display text characters in various sizes. A graphics-based HMI can display printable text characters of varying fonts and sizes and graphics shapes such as icons, bitmaps, or pictures. Using pictures instead of words or characters often greatly simplifies the operation of the HMI, making the HMI much more intuitive to use.

Some HMIs use touch screen displays while others use a membrane-style keypad. Membrane-style keyboards are best used in applications in which the keypad is likely to become dirty. Touch screen displays are placed over the HMI screen thus providing much more flexibility than typical membrane-style keypads. Because of this, switches can be created on a touch screen that appear only when needed.

The Weintek MT5xx HMIs are graphics-based touch screen HMIs. Before we get any further into the operation of these HMIs, it is necessary to define some terms that will be used throughout this manual.

Projects

The HMI has two basic segments of internal memory. The *code memory* contains the information required by the HMI that controls how it operates such as the features supported and how it communicates to a PLC. The HMI programmer does not have the ability to change code memory. The *project memory* pertains to all of the window screens created and any other features that the HMI programmer can create using the EZware-500 configuration software. Therefore, the term *project* is used to designate the file that is sent to the HMI from the EZware-500 software.

Objects

An *object* is any action that the HMI performs while it is communicating to the PLC. In order to get the operator interface terminal to 'do anything', you must program the HMI with objects. Objects perform actions such as display text or graphics, write a value to a PLC register, or display an alarm. Objects most often are graphics shapes that are to be displayed on the HMI screen. For example, a *Text Object* is used to display text on the HMI. But objects are also used to configure the HMI to perform some action. For example, a *PLC Control Object* tells the HMI to continuously monitor a PLC register that is used by the PLC to request a new window. Some objects can display a graphics shape on the HMI screen and perform some action. For example, a *Toggle Switch Object* creates a graphic object on the HMI that when pressed, activates a bit in the PLC.

Graphics Object

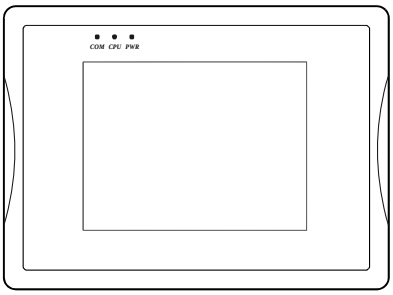
A *graphics object* is any text, icon, or picture that can be displayed on the HMI. Graphics objects are further defined by how they are composed or created. A *Text Object* is a graphics object that displays text on the HMI screen. A *Bitmap Object* is a graphics object that displays a bitmap on the HMI screen. Bitmaps are files stored in the HMI to display pictures. A *Shape Object* is a graphics object that displays a shape on the HMI screen. Shapes are also files stored in the HMI to display pictures. Shapes differ from bitmaps in that shapes are stored using a vector-based file format whereas bitmaps use a pixel-based file format. Each format has its advantages and disadvantages. We will not go into any more detail about bitmap objects and shape objects until later in this manual. For now, think of them as objects used to display pictures on the HMI screen. Finally, a *Group Object* is the most complex type of graphics object. It is a combination of other objects. Briefly, a group object consists of one or more objects that are 'grouped' together and stored as one object. A good example is a keypad, which is really a combination of several keys each designed to perform a specific task. When grouped together, a keypad can be stored as a group object for use in other projects or windows.

Windows

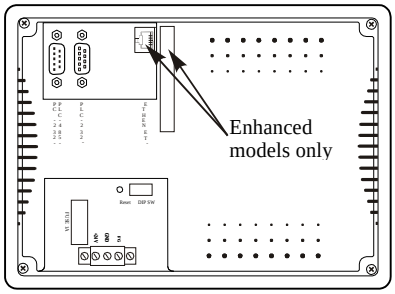
A *window* is a screen that can be displayed on the HMI. Windows can be full-sized to completely cover the HMI display or partially sized. Any partially sized window is usually referred to as a *popup window*. Windows can appear on the HMI display by a request from the PLC or by a press from the touch screen. Windows can be configured to any size. Once a window is displayed, it can be moved around the HMI display, removed from the display, or minimized to an icon. Windows can even overlap each other. Each window can display graphics objects and there is no limit to the number of graphics objects that can be placed on each window. The MT5xx is capable of storing up to 1999 windows , but the actual limit is determined by the total amount of memory used for the application. A more in-depth discussion of windows is covered in later chapters. For now, think of windows as screens that can be displayed on the HMI.

What is a MT5xx HMI?

The MT5xx of HMIs by Weintek are graphics operator interfaces designed to connect to PLCs in an industrial environment. The displays are covered with a 4-wire analog resistive touch screen designed for harsh industrial environments. The touch screen uses the latest in touch screen technology enabling the HMI programmer to create switches that are very fine in resolution. Unlike many other touch screen HMIs on the market, the MT5xx is not limited to a fixed number of cells in which switches can be created. The HMI programmer can create as many switches of varying sizes and shapes as he wishes, limited only by the total amount of memory available for the project.

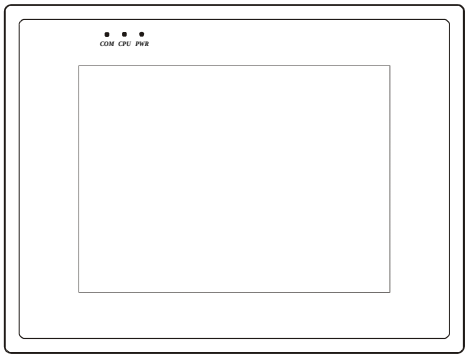


Front View

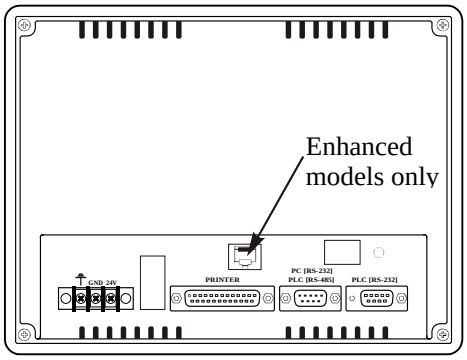


Rear View

MT506

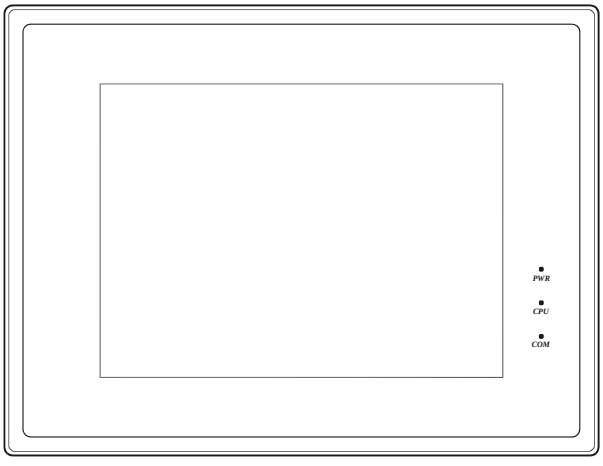


Front View

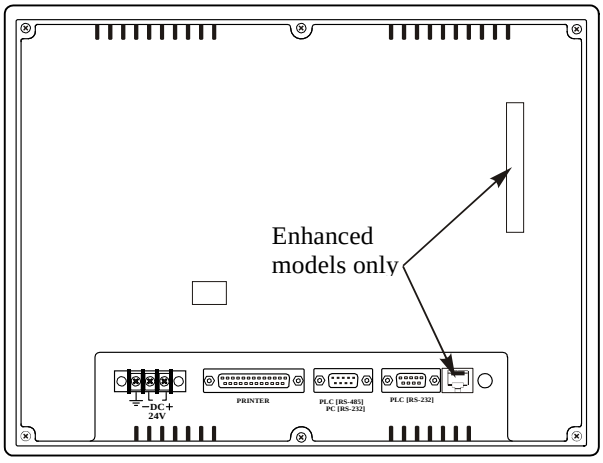


Rear View

MT508



Front View



Rear View

MT510

Three LED indicators are provided on the face of the MT5xx HMI to provide instant feedback to the HMI operator of the current operating condition of the HMI.

LED Indicator	Purpose
PWR LED (yellow)	indicates if power is applied to the HMI
CPU LED (green)	indicates if the HMI is operating correctly
COM LED (red)	indicates communications activity on PLC port

The MT5xx HMI has two serial ports, which provide a connection to a PLC using RS-232 or RS-485 communications and an RS-232 connection to a computer for programming. The serial ports also provide the ability to connect multiple HMIs in series to a single PLC port! The two serial ports also provide the ability to use the EZware-500 configuration software in Simulation Mode enabling the HMI programmer to test his project on the PC instead of downloading the project to the HMI.

The MT5xx is powered using +24VDC. The viewing level of the HMI display can be adjusted using a contrast switch locate on the back of the unit, or using Local Bits LB9091 and LB9092 for Contrast Up/Down. Finally, a reset switch is provided c the back of the HMI to reinitialize the HMI if an operational failure occurs. The MT5xx is designed for industrial environments and carries a NEMA 4 rating as well as CE compliance for noise immunity and emissions.

There are currently SEVEN models in the MT5xx.

Model	Display size	Resolution (pixels)	LCD Type	Brightness (cdm^2)	Contrast	Recipe/RTC Module	Ethernet & Compact Flash
MT506L	5.7”	320 x 240	4-shade blue mode STN	60	15:1	Standard	No
MT506TV	5.6"	320 x 234	256-color TFT	500	150:1	Standard	No
MT506TE (Enhanced)	5.6"	320 x 234	256-color TFT	500	150:1	Standard	Yes
MT508	7.7"	640 x 480	256-color TFT	400	250:1	Standard	No
MT508TE (Enhanced)	7.7"	640 x 480	256-color TFT	400	250:1	Standard	Yes
MT510TV	10.4”	640 x 480	256-color TFT	250	100:1	Standard	No
MT510TE (Enhanced)	10.4”	640 x 480	256-color TFT	250	100:1	Standard	Yes

Finally, the MT5xx are powered by a 200 MHz Intel XScale processor, making it one of the fastest HMIs available on the market. Even the most complicated graphics can be displayed quickly on the HMI screen. In addition, the HMI uses a proprietary algorithm to find the most efficient means of extracting data from the PLC. This lowers the amount of ‘bottleneck’ time due to the relatively slow communications baud rate of most PLCs. Therefore, the update rate to gather information from the PLC is very fast.

List of Features

The next chapter will guide you through the creation of your first project. Before you proceed, you may wish to read this brief list of some of the features offered in the MT5xx HMI.

Icon	Name	Description
	Bit Lamp	Creates a graphics object to reflect the current status of a PLC bit.
	Word Lamp	Creates a graphics object to reflect the current state of a multi-state PLC data register.
	Set Bit	Creates a touch screen graphics object that represents a two-state switch. When pressed it sets/resets a PLC bit.
	Set Word	Creates a touch screen graphics object that represents a multi-state switch. When pressed it can place a constant value in a PLC register or jog the value.
	Toggle Switch	Creates a touch screen graphics object that represents a two-state switch changing state (picture) based upon a PLC bit. When pressed, it can control another PLC bit.
	Multi-State Switch	Creates a multi-state touch screen graphics object that changes state (picture) according to the value in a PLC data register. When pressed, it sends a value(s) to another PLC register.
	Function Key	Creates a touch screen graphics object, which displays a window or edits a PLC register.
	Numeric Input	Displays a number stored in a PLC register. The number can be changed using a numeric keypad.
	Numeric Data	Displays a number stored in a PLC register
	ASCII Input	Displays ASCII characters stored in a PLC register. Characters can be changed using an alphanumeric keypad.
	ASCII Data	Displays ASCII characters stored in a PLC register
	Moving Shapes	Creates a multi-state graphics object, which changes state (picture) and position on the screen according to a value in a PLC register.
	Animation	Creates a multi-state graphics object, which changes state (picture) on the screen according to a value in a PLC register. The positions on the screen are predefined.
	Indirect Window	Configures the HMI to monitor PLC data registers or coils to display for a specific window popup by a PLC word address
	Direct Window	Displays a Window based on a bit in a PLC Register
	Alarm Displays	Creates alarms to display alarms sent from the Alarm Scan Object
	Trend Displays	Creates a trend graph. Samples data in a single or multiple 16-bit PLC register and plots the data on a time graph
	Bar Graph Displays	Creates a bar graph with alarm monitoring

	Meter Displays	Creates a scale meter
	Alarm Bar	Displays alarms detected by the Alarm Scan Object on a single horizontal scrolling line.
	Recipe Transfer	Transfers data to the specified PLC registers
	Event Displays	Displays messages according to ‘events’ that occur in the PLC
	Alarm Scan	Contains the data for detecting alarm conditions.
	System Message	Customizes the content of system-generated messages.
	PLC Control	Configures the HMI to monitor PLC data registers to display full window screens.
	Event Log	Monitors and records assigned events
	Data Transfer	Configures the HMI to periodically transfer data stored in one set of registers to another set of registers in the HMI or PLC.

Chapter 1 - Installation of HMIs

Before You Begin

Please read the following for proper handling of your new HMI.

Unpacking the Unit

Carefully unpack the HMI. Please read any instructions or cautions that appear on the shipping container. Check material in the container against the enclosed packing list. Weintek, Inc. will not accept responsibility for shortages ainst the packing list unless notified within 30 days. The equipment and its accessories were inspected and tested by Weintek before shipment; all of the equipment should be in good working order. Examine the equipment carefully; if any shipping damage is evident, notify the carrier immediately. You are responsible for claim negotiations with the carrier. Save the shipping container and packing material in case the equipment needs to be stored, returned to Weintek, or transported for any reason.

Managing Electrostatic Discharge

It is best NOT to remove the rear enclosure of the HMI. When the rear part of the enclosure is removed, the circuitry inside is exposed to possible damage by electrostatic discharge during handling. Minimize the possibility of electrostatic discharge by:

- Discharging personal static by grounding yourself prior to handling the HMI
- Handling the HMI at a static-free grounded workstation
- Connecting the frame ground (⏚) connector of the HMI to a clean earth ground
- Placing the HMI in an anti-static bag during transport

CE Compliance

The MT5xx HMIs have been tested to conform to European CE requirements per Council Directive 89/336/EEC. The European Union created these requirements to ensure conformity among products traded in those countries. Specifically, the MT5xx HMIs meet or exceed the noise emissions and immunity requirements as set forth in the EN50081 (Emissions) and EN50082 (Immunity) standards. These products are designed to withstand electrical noise in harsh industrial environments. They also conform to requirements that limit electrical emissions. However, this does not guarantee that the products will be totally immune from possible malfunction in cases where severe electrical noise occurs. Therefore, we strongly recommend that you follow the guidelines outlined in this chapter for proper wire routing and grounding to insure the proper operation of the MT5xx HMI.

NEMA Rating

The MT5xx HMIs are rated for NEMA 4/12 (indoor) or IP65 installations. This means that when the HMI is prope mounted to a panel or other enclosure, the front enclosure of the HMI will provide protection to the inside of the panel from splashing water, wind blown dust, rain, or hose-directed water. The HMI must be installed according to the instructions in this chapter to be properly sealed.

Environmental Considerations

The MT5xx is designed to operate in temperatures from 0-45° C. It is intended for indoor installations and not designed for outdoor applications. Avoid installing the MT5xx in environments with severe mechanical vibration or shocks. Do not install the HMI in enclosures with rapid temperature variations or high humidity. Either will cause condensation of water inside the device and eventual damage to the HMI.

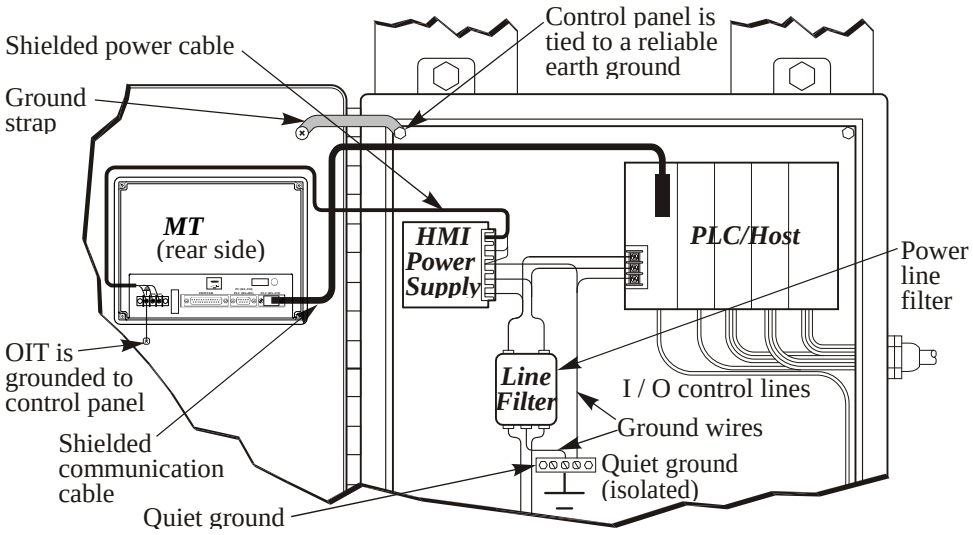
Safety Precautions

Please observe the following precautions when installing the MT5xx HMI. Failure to comply with these restrictions could result in loss of life, serious personal injury, or equipment damage.

	**Warning:** Do not operate the HMI in areas subject to explosion due to flammable gases, vapors, or dusts.
	**Warning:** Do not connect the HMI to an AC power source. You will cause permanent damage to the HMI.
	**Warning:** Do not attempt to use a DC power supply that does not meet HMI power requirements. You may cause malfunction or permanent damage to the HMI.
	**Warning:** Do not power the HMI with a DC power supply used for inductive loads or for input circuitry to the programmable logic controller. Severe voltage spikes caused by these devices may damage the HMI.

Control Panel Design Guidelines

Pay careful attention to the placement of system components and associated cable routing. These items can significantly enhance the performance and integrity of your control application.



Control Panel Example

Control Panel Grounding

The control panel should be connected to a good, high-integrity earth ground both for safety considerations and shielding purposes. Weintek cannot overemphasize the importance of good grounding. If you fail to use good grounding procedures during installation, sporadic malfunction of the HMI may occur:

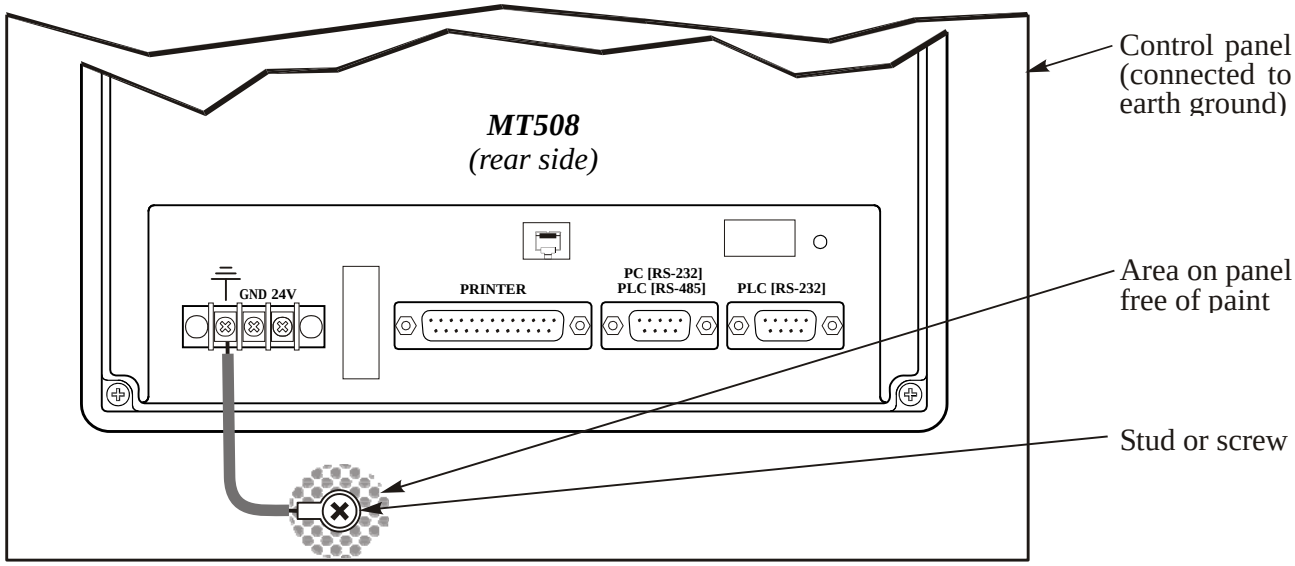
- Connect the HMI's chassis ground terminal to a reliable earth ground with a low-resistance path.
- Route all earth ground wires that lead from the HMI, the PLC, the power supply, and the line filter to a central earth ground point such as a barrier strip. This will ensure that no ground current from one device influences the operation of the other devices.
- Connect the HMI chassis ground terminal to the control panel door using a heavy-gauge short braided cable or ground wire to minimize resistance.
- Connect the power cable's shield wire to the HMI's chassis ground terminal.
- Connect the control panel to earth ground using a copper grounding rod close to the HMI and control panel.

Hinged doors on control panels do not provide a long-term electrical connection to the rest of the enclosure. Corrosion develops over time and prevents good electrical contract. For this reason, a separate wire braid should be installed from the hinged control panel to the rest of the enclosure.

Connect HMI Chassis Ground to Control Panel

To reduce the possibility of electrical interference, connect the chassis ground terminal of the HMI to a clean earth ground. If the control panel is metal, make sure it is properly grounded. Then connect a *short* heavy-gauge wire (#18 AWG) from the chassis ground terminal of the HMI to a mounting bolt on the control panel door. The mounting bolt must have good electrical contact to the control panel; scrape away any paint that may be covering the panel to provide a good connection.

If the control panel is made of a non-conductive material, it is essential that you connect the chassis ground terminal of the HMI to a clean earth ground point located close to the panel.



HMI Chassis Ground Connection

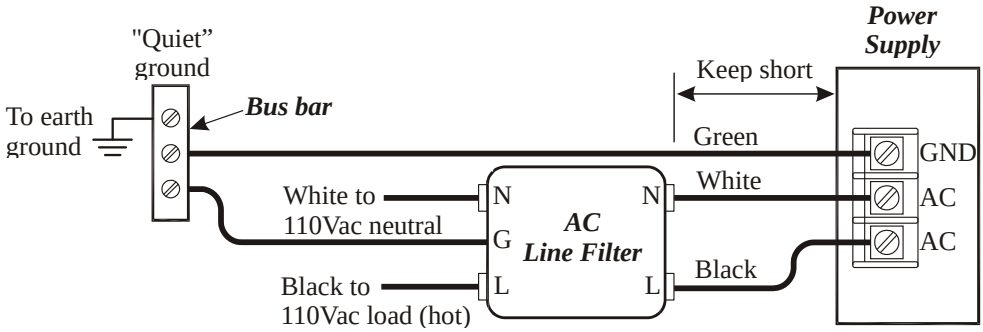
Power Supply Selection

The power supply used to power the HMI should provide an output of +24 VDC 5% measured at the HMI power terminal block. A 24VDC regulated power supply dedicated to the HMI is required (refer to Appendix A: Specifications for the input current requirements). Use a power supply with a 1.1 Amp output rating for the MT506 models and a 2.5 Amp output rating for the MT508 and MT510 models.

The power cable for the HMI should be 18AWG, 2-conductor wire with a shield drain wire and protective shield (foil or braid). The shield drain wire must be connected to earth ground at both ends of the cable. Please refer to the "Connect the HMI to Power" section for more information.

A power line filter installed at the AC input to the HMI power supply is highly recommended as a safeguard against conducted RF noise, which is often present on factory power lines. The wires connecting the output of the power line filter to the power supply should be kept as short as possible to minimize any additional noise pickup. The case of the power line filter should be connected to a quiet earth ground. The power line filter should have a current rating of at least 3 Amps with common mode and differential mode attenuation.

Do not use the power supply used to provide power to the HMI to power switching relays, solenoids, or other active devices.



Power Line Filter Connection

Cable Routing and Noise Immunity

Follow these guidelines when routing cable to the HMI:

- Always route the HMI communication cable and the power cable away from any AC voltage or rapidly switching DC control lines.
- Never bundle the HMI cables together with 120VAC power wires or with relay wiring.
- Try to keep at least 8 inches (20 cm) of separation between the HMI cables and other power wiring. If voltages greater than 120VAC are used in the system, greater separation is required.
- If the HMI cables must come near AC wiring, make sure they cross at 90 degrees.
- Run AC power wires in a separate grounded conduit to reduce electrical noise interference.
- Keep the cable lengths for the HMI as short as possible. Do not coil excess cable and place it next to AC powered equipment.
- Cover any equipment used in the enclosure that operates at high frequency or high current levels with a grounded metal shield.

Installation

It is necessary to follow all installation procedures described in this chapter for electrical noise immunity and CE compliance.

Your Weintek HMI is designed to connect easily to your PLC. External rear connectors provide quick connections for power, communications and programming wiring.

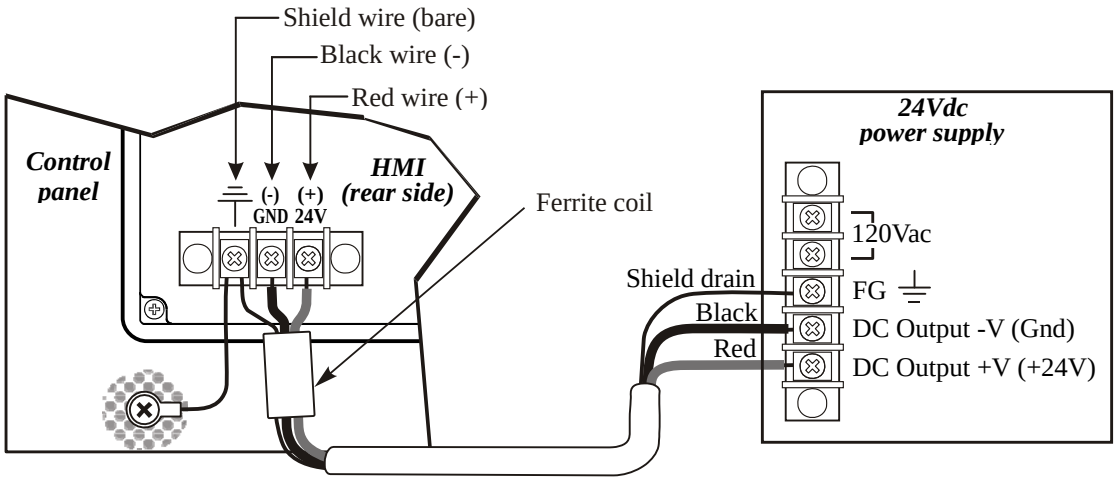
There are two serial ports on the rear of the HMI. Both of these are D-subminiature 9-pin connectors. Use the female connector for RS-232 communications to a PLC. Use the male connector for RS-485 communication to a PLC and RS-232 communications to a PC.

Use the supplied separate 3-position terminal block to provide power to the HMI.

Connect the HMI to Power

The power cable for the HMI should be 18AWG, 2-conductor wire with a shield drain wire and protective shield foil. You may buy cable P/N 6030-0009 by the foot from Weintek to make these.

Always run the DC ground wire directly back to the signal return of the power supply. *Do not use the chassis ground wire as your signal return!* Weintek recommends using an axial ferrite coil with each MT510H HMI to further reduce the electrical interference that may be conducted on the power lines. Thread the positive and negative wires of the power cable through the ferrite coil so that the coil is no more than two inches from the HMI's power input.



HMI Power Inputs

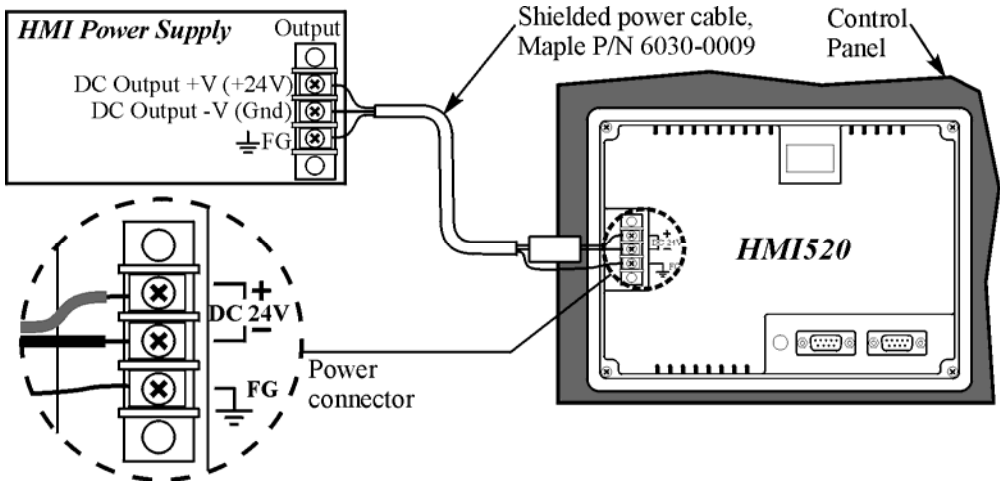
To connect the HMI to power:

1. Connect the power cable to the HMI
   a. Strip the power cable shield to expose 2" of the black and red wires.
   b. Strip about ¼" of insulation from the black and red wires.
   c. Thread the black and red wires through the ferrite core. The shield wire must be outside.
   d. Connect the red wire to the DC positive (+) input of the HMI power terminal.
   e. Connect the black wire to the DC negative (-) input of the HMI power terminal.
   f. Connect the power cable shield wire to the HMI power terminal's chassis ground input.
2. Route the power cable to the HMI power supply. The power cable should not be any longer than necessary.

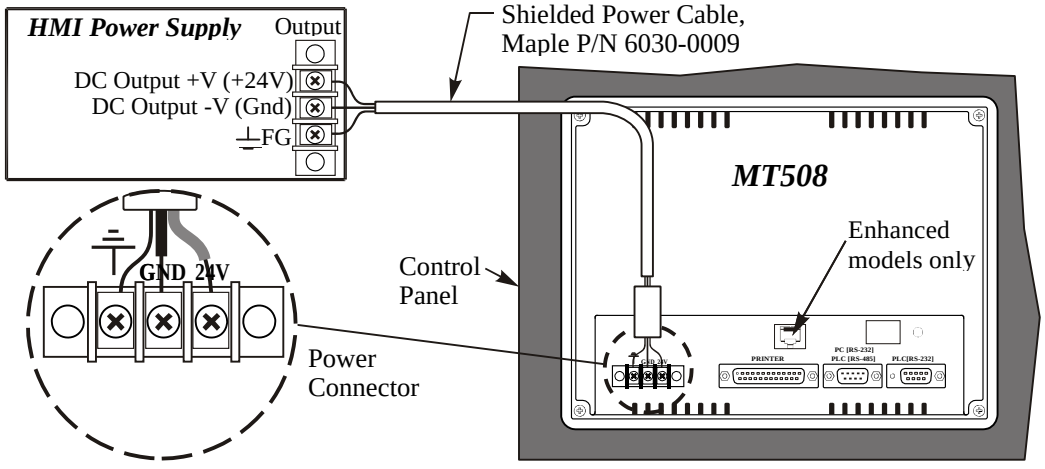
3. Install the power supply wires as follows (with colors shown for Weintek cable P/N 6030-0009):

Color	Power Supply	MT506	MT508	MT510
Red	+Output/+24 Vdc	+dc 24V	24V	+dc 24V
Black	-Output/+24 Vdc return	-dc 24V	GND	-dc 24V
Shield	Case ground	FG	⏚	⏚

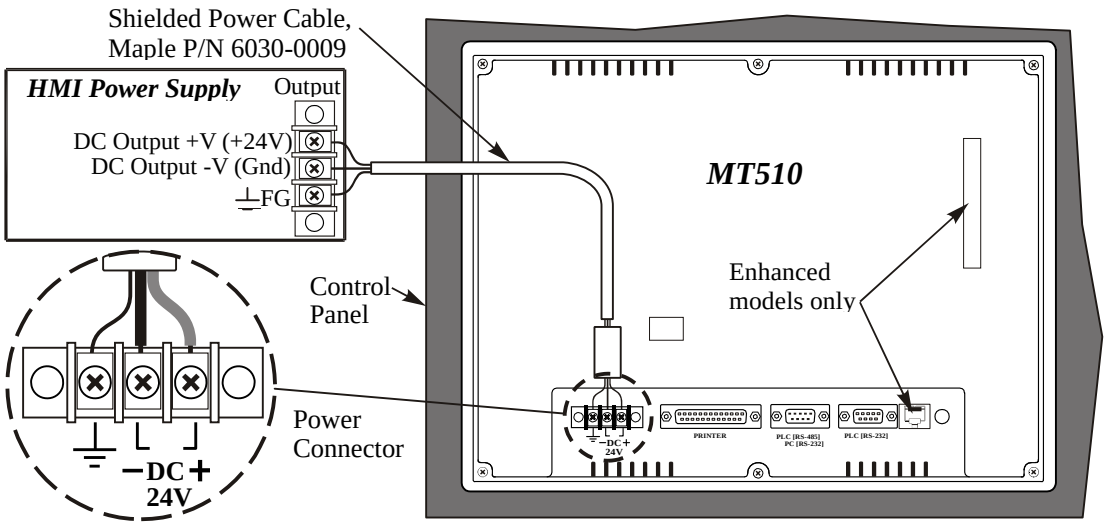
The power connector on the MT5xx is a terminal block with wire clamps. Lugs are not required.



MT506 Power Wiring



MT508 Power Wiring



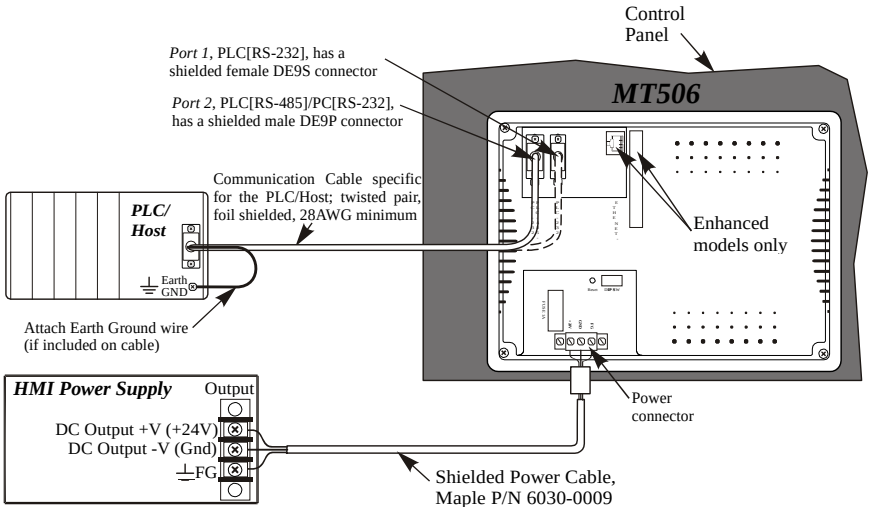
MT510 Power Wiring

Connect the HMI to the PLC

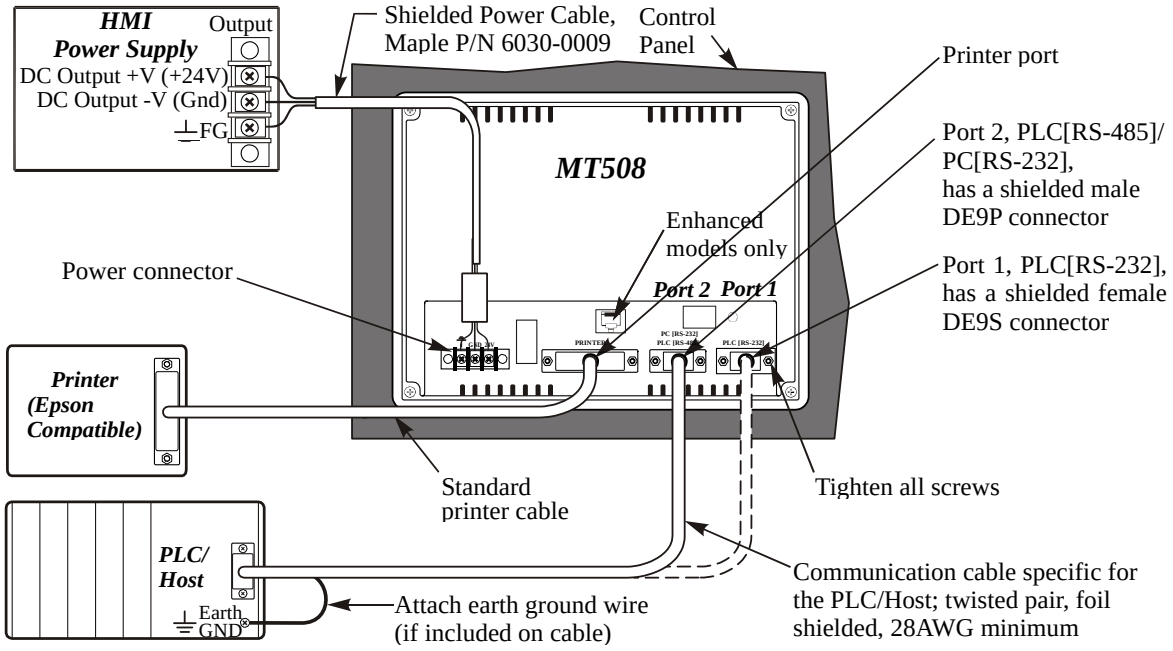
Each PLC supported by Weintek has its own wiring requirements. Weintek offers HMI-to-PLC communication cables for most PLCs that are built to any length and tested for high reliability. Most cables are available for next-day shipment from Weintek. Components and instructions necessary to construct your own HMI-to-PLC communications cables are also available. Refer to Weintek' *Technical Note 1061* or visit our W site,

Port 1 PLC (RS-232)			Port 2PLC (RS-485), PC (RS-232)	
Pin #	**Function**		**Pin #**	**Function**
1	(no connection)		1	RXD-
2	TXD		2	RXD+
3	RXD		3	TXD-
4	(no connection)		4	TXD+
5	GND		5	GND
6	(no connection)		6	(no connection)
7	(no connection)		7	TXD
8	(no connection)		8	RXD
9	(no connection)		9	(no connection)

Pinout for the HMI Ports



Wiring the MT506 to a PLC



Wiring the MT508/MT510 to a PLC

STEPS:

1. Connect the “HMI” end of the communication cable into either the RS-232 port or the RS-485 port as required for your application (HMI housing is marked).
2. Tighten the two cable screws at each end to ensure shield ground path.
3. Route the communication cable to the PLC. Refer to the “HMI Cable Routing” section for more information.
4. Connect the “PLC” end of the cable to the PLC and tighten the cable screws.
5. Connect the green shield wire from the cable to earth ground ( ) on the PLC. If this wire is not present, make the ground connection inside the PLC connector.
6. The MT508/550 has a printer port and can be connected to an Epson compatible printer.

Panel Preparation

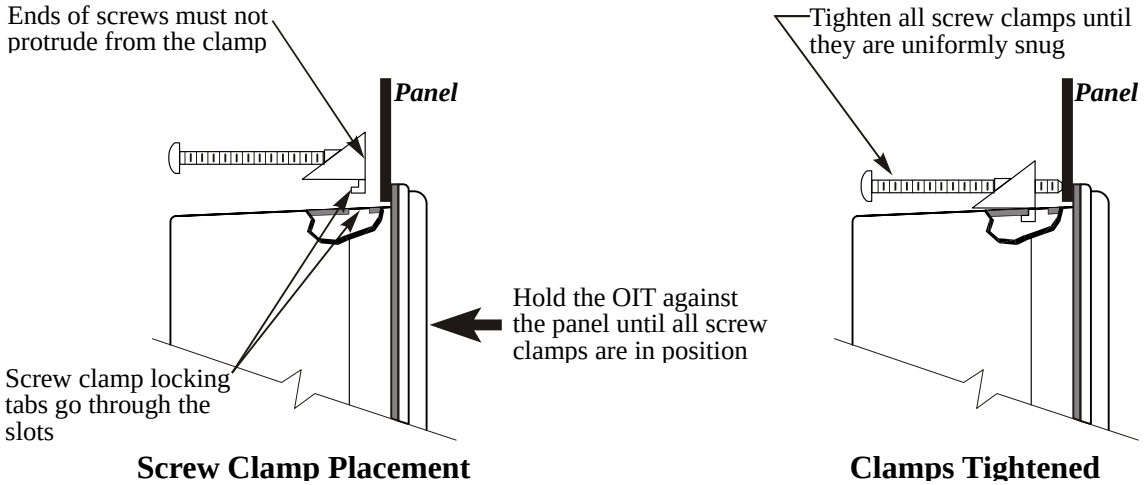
A metal panel or mounting surface with a minimum thickness of 15 gauge (0.059 inch/3.3mm) if cold-rolled steel or hardened steel, or 10 gauge (0.101 inch/2.6mm) if aluminum alloy (6061-T6 preferred) is required. Thinner panels or surfaces may bow between the mounting clamps and not form a seal with the gasket.

The area of the panel or mounting surface where the gasket comes into contact must be flat and free of scratches, pits, and other features that prevent the gasket from sealing properly. If the panel or mounting surface is not uniform, thick, flat, stiff, or smooth enough, then a sealant such as silicone may be required.

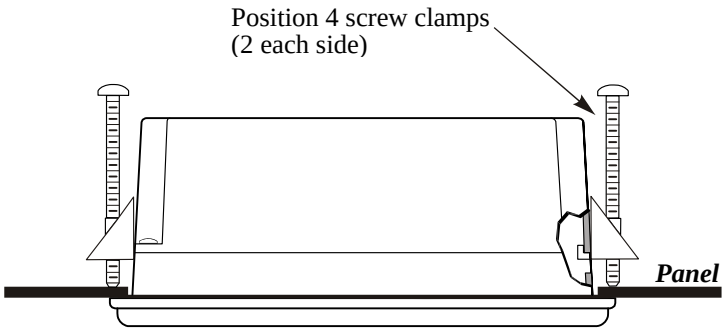
WARNING: *The HMI requires a stiff, flat, smooth mounting surface free of blemishes to seal properly to NEMA 4.*

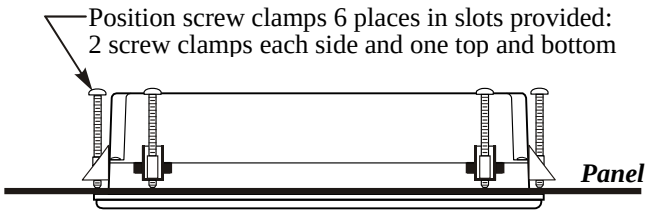
The diagrams in Appendix C show the dimensions of the panel cutout required for proper installation of the MT506, MT508, and MT510 models.

Clean and deburr the panel cutout before the HMI is installed.



Installing Screws on the HMI





Side View

STEPS:

1. Prepare the four screw clamps for the MT506/530 or the six screw clamps for the MT510 by positioning the metal brackets at the midpoints of the screws. Position the screws so that the ends don't protrude from the plastic portions.
2. Set the HMI in the panel cutout and hold it in place until all clamps are in position.
3. Tighten the screw clamps until all are uniformly snug.

CAUTION: Do not over-tighten the screws beyond snugness, or you may damage the housing.

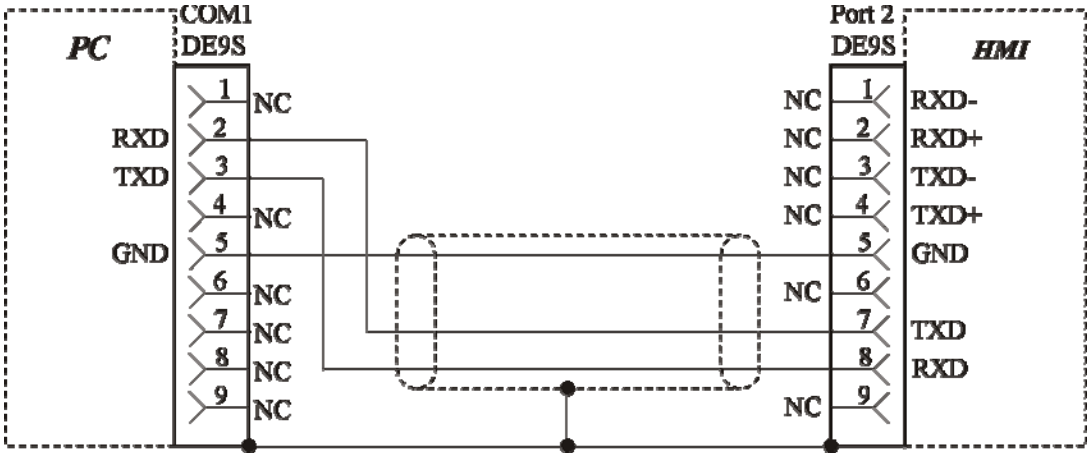
REINSTALLATION: Because the gasket will take a "set" to the panel, be sure to reinstall the HMI to the same panel cutout when a NEMA 4 seal is required. For best results, also replace the gasket itself.

Configuration Wiring

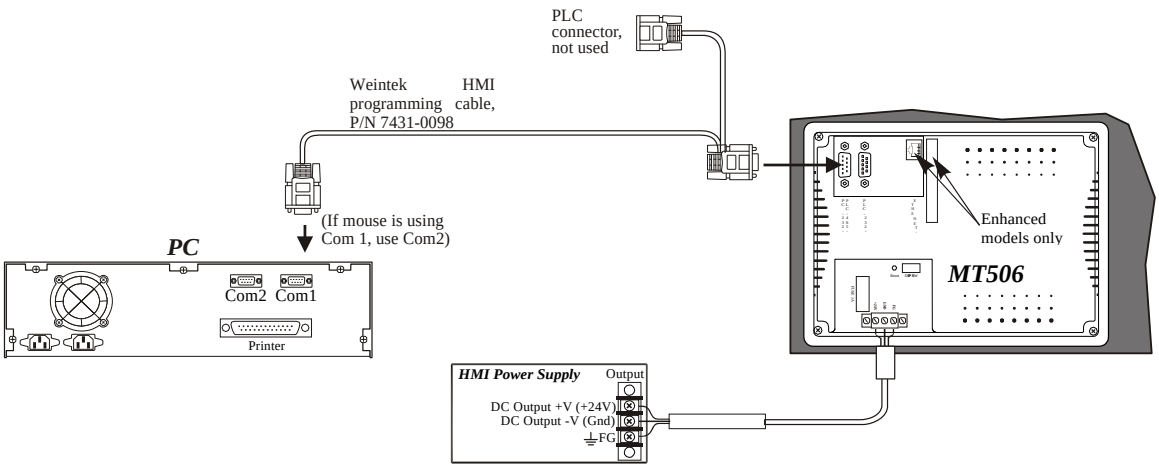
The HMI must be configured for a particular protocol before use. The *EZware-500* software (used on a PC with Windows 95 or higher) is used for configuring the HMI. For detailed instructions on installing and using the software, please refer to the software documentation section of this manual.

Connect the HMI to the PC for Configuration

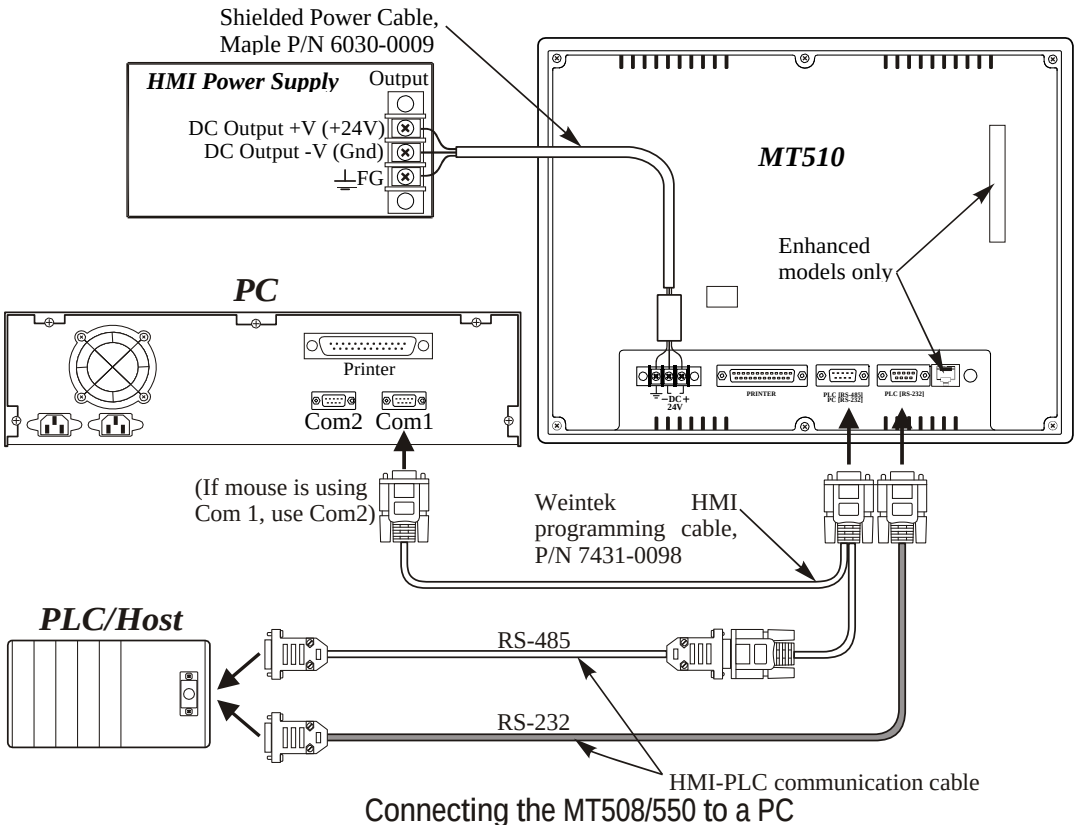
To configure the HMI using Maple System's configuration software, you will need the HMI Configuration Y-adapter Cable, Maple P/N 7431-0098. Connect the end marked "HMI" into Port 2 on the HMI, marked "PLC[RS-485]/ PC[RS-232], and connect the end marked PC into the proper COM port on your PC. The end marked "PLC" is not used here. See the figure below for serial port pin assignments and the next two figures for connecting the MT5xx to a PC.



HMI to PC Serial Port Pin Assignments



Connecting the MT506 to a PC



Connecting the MT508/550 to a PC

Factory Configuration

Each HMI arrives from the factory with a demo project file that illustrates some of the most popular features of the HMI. Please follow the directions enclosed in *Chapter 2, Creating Your First Project,* to configure your HMI for the PLC that you are using.

Each HMI has a potentiometer located on the back for adjusting the level of contrast on the display. It is set for optimum clarity at the factory but you can change the setting if necessary.

The HMI also has a black reset pushbutton and a four position DIP switch located through an access hole on the back of the enclosure. The reset switch can be used to reinitialize the HMI if the HMI malfunctions. Only Dip switch 1 and 2 have functionality. Dip switch 1 puts the HMI into Touch Calibration mode. Dip switch 2 forces the HMI into download mode, and needs to be set for downloading from a Compact Flash card. For normal operation, all of the DIP switches should be set to the OFF position.

The touch screen of the HMI is fully calibrated before it leaves the factory so you shouldn't need to adjust it. However, with time the touch screen may need to be recalibrated.

▶ To calibrate the touch screen

1. Connect the HMI to your computer for programming.
2. From the **Start** menu, click **Programs,** then **Weintek**, then **EZware-500**, then the **EasyManager** program.
3. Click **Jump To Touch Adjust**. The HMI screen should change to calibration mode.
4. After the HMI displays a crosshair cursor, you are prompted to touch the cursor for:

a. Top left position
b. Top right position
c. Bottom left position
d. Bottom right position

5. The HMI screen displays two rectangular objects. To determine whether the touch screen is properly calibrated, place and hold your finger somewhere on the touch screen other than on the rectangular spots. The crosshair cursor should appear directly under your finger. If needed, press the left rectangle to repeat the calibration. Press the right rectangle to end calibration.
6. Once you have completed calibration, press the touch screen again to return to normal application mode.

Chapter 2 - Creating Your First Project

Often the best way to learn about new software is to just jump right in. This chapter will step you through the process of installing the EZware-500 configuration software and then using the software to create a sample project that can be downloaded to your HMI. We won't go into much detail as to how each feature works. The purpose of this chapter is only to provide you with an overview of the process of creating a functional HMI that can communicate to a PLC. For our sample project, we will configure the HMI using the MemoryMap_Master protocol but you may feel free to select whichever protocol driver you intend to use.

By the end of this chapter, you should be able to:

- Install EZware-500 configuration software.
- Create a sample project with two windows and several graphics objects.
- Save a project, compile a project and download the project to the HMI.
- Verify that the HMI is functioning properly.

Before You Begin

Before you install EZware-500, make sure your computer meets the following minimum system requirements:

- Pentium-based 90MHz or higher processor
- 16 MB of RAM (more memory improves performance)
- 10 MB available hard disk space
- VGA or higher-resolution monitor set for 256 color 800x600 pixel mode
- Microsoft Mouse or compatible pointing device
- One available RS-232 port
- Microsoft Windows 95, 98, NT, XP or higher

Connecting HMI to Computer

Before you start your first project, the HMI should be connected to the computer so that the project can be downloaded after creating it. You should also connect the PLC that you are using to the HMI so that you can test the operation of the HMI after you have finished creating this sample project.

To connect your HMI to the computer

1. Connect a +24VDC power supply to the HMI.
2. Connect the Y-adapter programming cable (Maple P/N 7431-0098) to the computer and HMI.
   - Connect the end marked **HMI** to the HMI port labeled **PC[RS-232]**.
   - Connect the end marked **PC** to the COM port of the computer.
3. Apply power to the HMI.

Changing the PC COM Port used by EZware-500

1. EZware-500 is initially configured to use Com Port 2:
2. In Windows, click the Start button.
3. Select Programs.
4. Select EZware500.
5. Select EasyManager.

6. In the top left drop-down list box, select COM 1.
7. Choose **EXIT.**

▶ To connect your PLC to the HMI

1. Weintek produces PLC communications cables, which will connect the HMI to most of the PLCs available. The cables can be manufactured to any length you may require. A listing of all the PLC cables Weintek offers can be found on our website at
2. If you have decided to make your own cable, make sure that it is wired correctly for your particular PLC. Weintek provides cable wiring diagrams to guide you through the construction of your cable. These can also be found on our website at www.maple-systems.com/cables.htm.
3. Connect the PLC communications cable from the serial port on your PLC to the appropriate serial port on your HMI.
   - If you are using RS-232 communications, then connect the HMI end of the cable to the HMI port labeled **PLC[RS-232]**.
   - If you are using RS-485 communications, then connect the HMI end of the cable to the HMI port labeled **PLC[RS-485]**. If you wish to use the Y-adapter programming cable, then connect to the cable end that is marked **PLC**.
4. Apply power to the PLC. The MT5xx HMI is programmed at the factory with a sample demo that shows some features of the HMI. The HMI should display a sample startup screen. The HMI will now accept a new project from EZware-500.

Starting EZware-500

Before you can create a sample project, you must start the configuration software. The EZware-500 software has three main applications:

- **EasyBuilder** – used to create the project downloaded to the HMI.
- **EasyManager** - used to place the HMI into different operating modes.
- **EasyASCIIFontMaker** - used to edit the text fonts that display characters on the HMI.

▶ To start the EZware-500 software

1. From the Windows Task Bar, click the **Start** button, point to **Programs**, and then click the **EZware-500** folder.
2. Click **EasyManager** to start the EasyManager application.
3. Select the appropriate *COM port* on your computer.
4. Select a baud rate of **38400.**

After you have successfully downloaded a project to the HMI using this baud rate, try using 115200 to decrease the required download time.

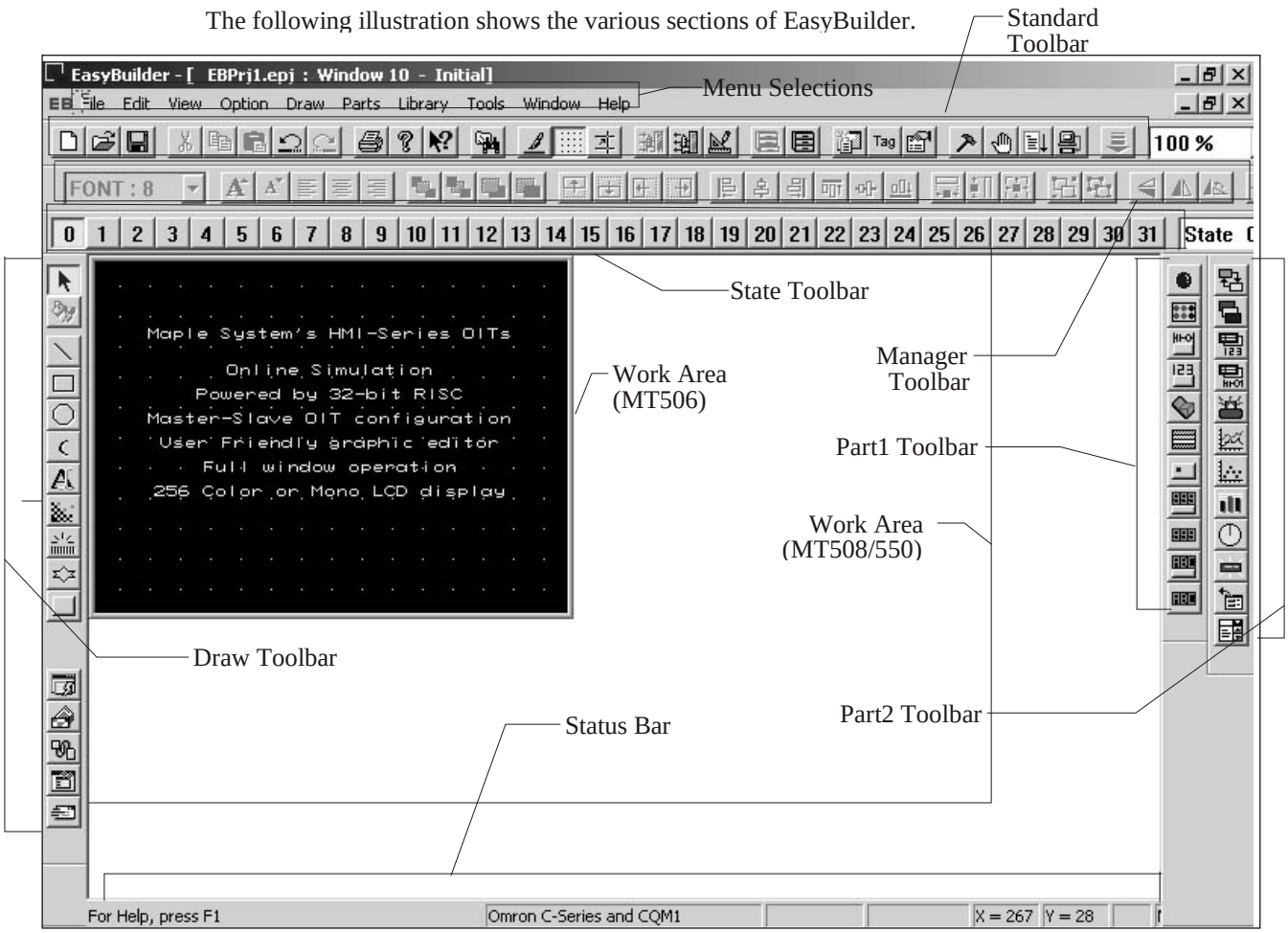
5. Click **Project Download/Upload**.
6. Click **Complete Download/Upload**.

After you have downloaded your first project to the HMI, you can switch to Partial Download/Upload to decrease the required download time.

7. On the EasyManager dialog box, click **EasyBuilder**.
8. The Welcome to EasyBuilder dialog box appears. Select the correct *HMI model.*
9. Click **OK** to display the main screen of EasyBuilder.

*If the main screen of EasyBuilder is displayed after you click **EasyBuilder** from the EasyManager dialog box, select the correct model of HMI by clicking **New** from the File menu.*

The following illustration shows the various sections of EasyBuilder.



Creating a Sample Project

This section walks you through the creation of an EasyBuilder project named EBPrj1. Once downloaded to the HMI, this basic configuration allows the HMI to connect to the PLC, display a startup screen, and display a screen containing one PLC register monitor when a switch on the startup screen is pressed.

Although we strongly recommend that you perform the following steps to create this sample project, the project is already included in your EasyBuilder software with the following filenames:

MT506M.EPJ
-sample project for the MT506M

MT506C.EPJ
-sample project for the MT506C

MT506T.EPJ

-sample project for the MT506T

MT508C.EPJ
-sample project for the MT508C

MT508T.EPJ

-sample project for the MT508T

MT510M.EPJ
-sample project for the MT510M

MT510H.EPJ
-sample project for the MT510H

Setting the System Parameters

Whenever you begin a new project, you should always set the system parameters before you create any windows. System parameters determine the basic operating conditions of the HMI such as what type of PLC it is connecting to.

To edit the System Parameters

1. Click the **Edit** menu on the main screen of EasyBuilder.
2. At the bottom of the menu, click **System Parameters**. The Set System Parameters dialog box appears.
3. The dialog box has six tabs: PLC, General, Indicator, Security, Editor and Hardware. Select the **PLC** tab.
4. Select the PLC that you are using or **MemoryMap_Master** from the PLC type pull-down box.
5. Select **RS485** from the Serial port I/F box.
6. Select **8 bits** from the Data bits box.
7. Select **1 bit** from the Stop bits box.
8. Select **19200** from the Baud rate box.
9. Select **Odd** from the Parity box.
10. Select **0** for the HMI station No. and PLC station No. Boxes.
11. Select **Disable** from the Multiple HMI box.
12. Select **3.0** from the PLC time out constant box.
13. Select **0** from the PLC block pack box.
14. Now click on the **General** tab.
15. Select **10** for the Startup window No. Box.
16. Click **OK** to return to the EasyBuilder main screen.

The communications parameters selected above were selected at random. Please enter the PLC type and communications parameters that match your PLC. For more information, consult your PLC operations manual or Weintek Controller Information Sheets available on our website at www.maple-systems.com .

Creating a Startup Window

We will configure Window #10 as the startup screen. The HMI can store up to 1999 predefined windows but screens 0-9 are reserved. Therefore, the first window screen that is available for configuration is Window #10. By default, this is the startup screen when you begin a new project. This section will show how to place text in the window and how to create two function keys that will open and close Window #11.

To place text on Window#10

1. From the Draw menu, click **Text**. The Create Text Object dialog box appears.
2. Click the pull down box from the Color box. The Color dialog box appears.
3. Click on the white color box, then click **OK**. The color box should reflect the color that you have chosen.
4. Select **16** in the Font box.
5. Select **Left** in the Align box.
6. Double-click the word 'text' in the Content box and type "**This is the Startup Screen!**."
7. Click **OK**.
8. On the main screen of EasyBuilder, you will see a white rectangle outline that is attached to your cursor in the work area. This represents the text box just created. Center the rectangle somewhere on the top third portion of the work area and then click. The text box "This is the Startup Screen!" should appear.

To create a function key on Window#10

1. From the Parts menu, click **Function Key**. The Create Function Key Object dialog box appears.
2. Type **On Button** in the Description box.
3. Click Popup Window.
4. Type **11** in the Window No. Box.
5. Click the **Shape** tab.
6. Click **Use shape**, then click **Shape library**. The Shape Library dialog box appears.
7. Click **button1** in the Shape library box.
8. Scroll through the selections to selection 23. Click the shape, then click **OK**. The shape should appear in the Shape tab.
9. Click the **Label** tab.
10. Click the pull down box from the Color box. The Color dialog box appears.
11. Click on the black color box, then click **OK**. The color box should reflect what you have chosen.
12. Select **16** in the Font box.
13. Select **Left** in the Align box.
14. Type **ON** in the Content box.
15. Click the **Use Label** box.
16. Click **OK**.
17. On the main screen of EasyBuilder, you will see a white square outline that is attached to your cursor in the work area. This represents the function key just created. Center the square somewhere on the bottom left portion of the work area and then click. The function key should appear.
18. Double-click on the function key. The Function Key Object's Attribute box appears.
19. Click on the **Profile** tab.
20. Enter **47** for the X position. Enter **168** for the Y position.
21. Enter **62** for the Width. Enter **49** for the Height.
22. Click **OK**.
23. On the main screen of EasyBuilder, you should see the function key that you just created move and/or change size. This function key is used to display Window #11.

To create a second function key on Window#10

1. We could create the second function key by repeating the steps for the first function key. Instead, however, we will take advantage of the first function key and copy it.
2. Click the first function key to highlight it.
3. From the Edit menu, click **Copy**.
4. From the Edit menu, click **Paste**. A copy of the function key appears in the upper left corner of the work area. Deselect the function key by clicking on any blank space in the work area.
5. Double-click on the second function key. The Function Key's Object Attribute dialog box appears.
6. In the General tab section, change the Description to **Off Button**.
7. Click on **Change Window**. Enter **10** for Window number
8. Click on the **Label** tab. Type **OFF** in the Content box.
9. Click on the **Profile** tab.
10. Enter **196** for the X position. Enter **168** for the Y position.
11. Click **OK**.
12. On the main screen of EasyBuilder, you should see the second function key that you just created move to the lower right hand side of the work area. This function key is used to close Window #11.

You have finished configuring your first window. It should look something like the picture below:

Creating a Popup Window

We will configure Window #11 as a popup window. Up to six popup windows can be displayed on a full screen window. The windows may overlap each other or may be minimized to an icon. This section will show how to create a scale and a numeric register that displays the current value of the scale. You will also create an increment and decrement key to change the value in the scale meter.

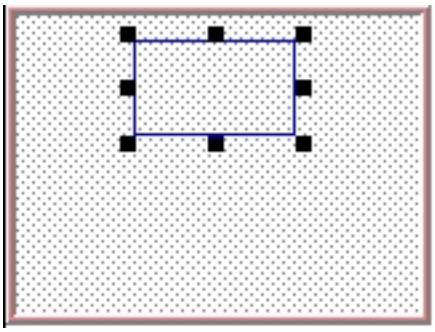
To create Window#11

1. From the Window menu, click **Open Window**. The Open Window dialog box appears.
2. Click **New Window**. The Select Window Style dialog box appears.

3. Click **Base Window**. The Window Setting dialog box appears.
4. Enter **79** for the X position. Enter **58** for the Y position.
5. Enter **160** for the Width. Enter **120** for the Height.
6. Click the pull down box from the Background Color box. The Color dialog box appears.
7. Click on the white color box, then click **OK**. The color box should reflect the color that you have chosen.
8. Click **OK**.
9. Select **Window_11** in the Open Window dialog box.
10. Click **Open.** The main screen of EasyBuilder should reappear with Window #11 displayed in the work area.

To create a rectangle on Window#11

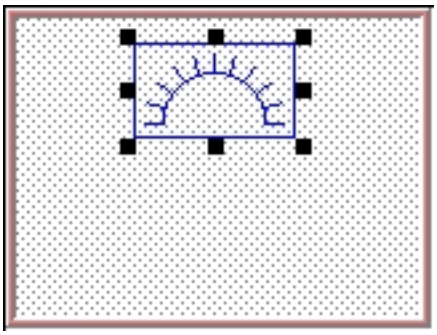
1. From the Draw menu, click **Rectangle**. The Attributes dialog box appears.
2. Click on Interior Filled.
3. Click the pull down box from the Interior box. The Color dialog box appears.
4. Click on the white color box, then click **OK**. The color in the Interior box should reflect the color that you have chosen.
5. Click on an area somewhere inside Window #11. Create a rectangular shape by moving the mouse cursor. Click again to permanently create the rectangle.
6. From the Edit menu, click **Select**.
7. Double-click somewhere on the rectangle. The Rectangle Object's Attribute dialog box appears.
8. Click the **Profile** tab.
9. Enter **48** for the X position. Enter **13** for the Y position.
10. Enter **61** for the Width. Enter **36** for the Height.
11. Click **OK**.
12. On the main screen of EasyBuilder, you should see the rectangle that you just created move and/or change size. This rectangle is used as a backdrop to the scale meter.



To create scale lines on Window#11

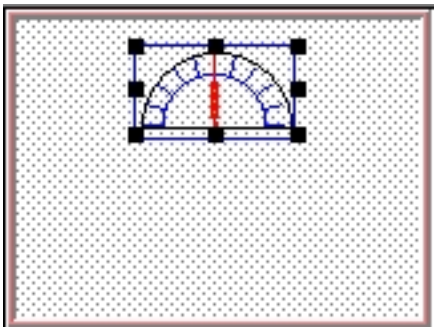
1. From the Draw menu, click **Scale**.
2. Click on an area somewhere inside Window #11. Create a scale shape by moving the mouse cursor. Click again to permanently create the scale lines.
3. From the Edit menu, click **Select**.
4. Double-click somewhere on the scale lines. The Scale Object's Attribute dialog box appears.
5. In the Scale Style box, click **Up**.
6. Type **10** in the Division box.
7. Type **8** in the Meter length box.
8. Click on the **Profile** tab.
9. Enter **51** for the X position. Enter **16** for the Y position.

10. Enter **56** for the Width. Enter **28** for the Height.
11. Click **OK**.
12. On the main screen of EasyBuilder, you should see the scale that you just created move and/or change size. This scale should be enclosed by the rectangle.



To create a meter display on Window#11

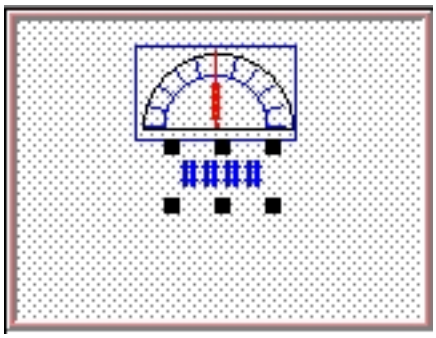
1. From the Parts menu, click **Meter Display**. The Create Meter Display Object dialog box appears.
2. Type **Meter Display** in the Description box.
3. Click the **Meter Display** tab.
4. In the Style 2 Indicator box, click **Up half**.
5. In the Value Span box, enter **100**.
6. Click **OK**.
7. On the main screen of EasyBuilder, you will see a white square outline that is attached to your cursor in the work area. This represents the meter display just created. Click to place the meter display in Window #11.
8. Double-click on the meter display. The Meter Display Object's Attribute dialog box appears.
9. Click on the **Profile** tab.
10. Enter **51** for the X position. Enter **16** for the Y position.
11. Enter **56** for the Width. Enter **28** for the Height.
12. Click **OK**.
13. On the main screen of EasyBuilder, you should see the scaled meter display that you just created move and/or change size. The scaled meter display is now complete.



To create a numeric register on Window#11

1. From the Parts menu, click **Numeric Data**. The Create Numeric Data Object dialog box appears.
2. Type **Numeric Data** in the Description box.

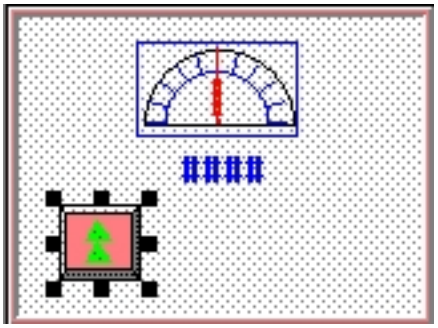
3. Click **OK**.
4. On the main screen of EasyBuilder, you will see a white rectangle outline that is attached to your cursor in the work area. This represents the numeric register just created. Click to place the numeric register on Window #11 somewhere underneath the scale meter. Four pound signs "####" should appear.



To create an increment key on Window#11

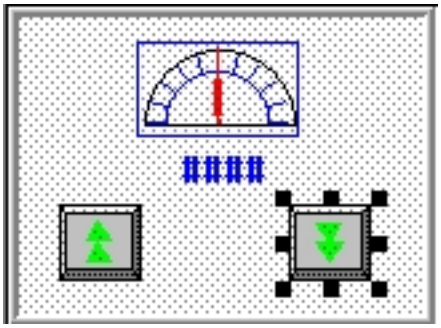
1. From the Parts menu, click **Set Word**. The Create Set Word Object dialog box appears.
2. Type **Increment Button** in the Description box.
3. Click on the pull down box of the Set Style Attribute and select **JOG++**.
4. Enter **1** in the Inc. value box.
5. Click on the pull down box of JOG Delay and select **0.5 second**.
6. Enter **100** in the Attribute Upper limit box.
7. Click the **Shape** tab.
8. Click **Use shape**, then click **Shape library**. The Shape Library dialog box appears.
9. Click **Select Lib…** The Open dialog box appears.
10. Select **button3.slb** from the list of shape libraries. Then click **Open**.
11. Click **button3** in the Shape library box.
12. Scroll through the selections to selection 21. Click the shape, then click **OK**. The shape should appear in the Shape tab.
13. Click **OK**.
14. On the main screen of EasyBuilder, you will see a white square outline that is attached to your cursor in the work area. This represents the increment key just created. Click to place the increment key on Window #11.
15. Double-click on the increment key. The Set Word Object's Attribute dialog box appears.
16. Click on the **Profile** tab.
17. Enter **19** for the X position. Enter **74** for the Y position.
18. Enter **31** for the Width. Enter **29** for the Height and click **OK**.

19. On the main screen of EasyBuilder, you should see the increment key that you just created move and/or change size. The increment key is now complete.

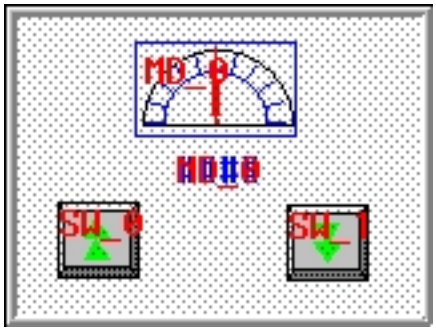


To create a decrement key on Window#11

1. As with the second function key of Window #10, we are going to create the decrement key by copying the increment key and then making changes to the attributes.
2. Click the increment key to highlight it.
3. From the Edit menu, click **Copy**.
4. From the Edit menu, click **Paste**. A copy of the increment key appears in the upper left corner of the work area. Deselect the increment key by clicking on any blank space in the work area.
5. Double-click on the second increment key. The Set Word Object's Attribute dialog box appears.
6. In the General tab section, change the Description to **Decrement Button**.
7. Click on the pull down box of the Set Style Attribute and select **JOG—**.
8. Enter **0** in the Attribute Bottom limit box.
9. Click the **Shape** tab.
10. Click **Shape library**. The Shape Library dialog box appears.
11. Scroll through the selections to selection 22. Click the shape, then click **OK**. The shape should appear in the Shape tab.
12. Click on the **Profile** tab.
13. Enter **105** for the X position. Enter **74** for the Y position.
14. Click **OK**.
15. On the main screen of EasyBuilder, you should see the decrement key that you just created move to the lower right hand side of the work area.



The following illustration shows how the popup window looks:



You have now done your part in creating this sample project. It is now time for EZware-500 to do its part.

Finishing Up

There are still a few steps, which must be completed before you can test your first project. In this section, you will:

- save the project onto your computer hard drive
- compile the project into a format that can be understood by the HMI
- download the project to the HM
- verify that the HMI operates as expected
- exit the EZware-500 software

If you haven't already done so, now would be a good time to connect the HMI to the computer and to connect the PLC that you are using to the HMI. For more information, consult the first part of this chapter or see "Installation of HMIs" later on in this manual.

Saving your first project

1. From the File menu, click **Save As**. The Save As dialog box appears.
2. In the File name text box, type **EBPrj1**.
3. Click **Save**. The file is saved onto your computer hard drive and the main screen of EasyBuilder reappears.

Compiling your first project

1. From the Tools menu, click **Compile**. The Compiling dialog box appears.
2. Click **Compile**. EasyBuilder will compile your project and display error results.
3. If no errors occur, click **Close**. The main screen of EasyBuilder reappears. If errors have occurred, repeat the steps in the **Creating a Sample Project** section.

Downloading your first project

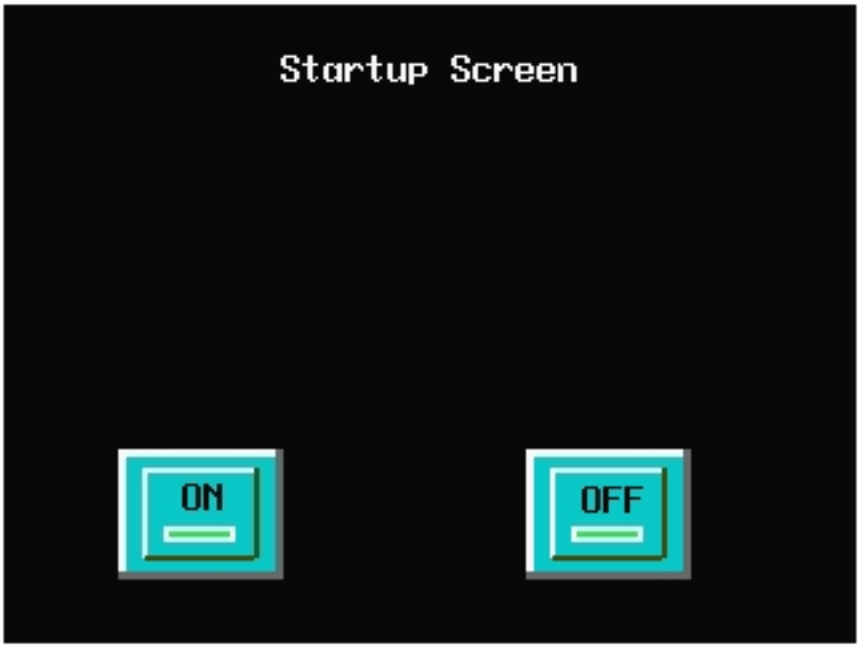
1. Apply power to your HMI.
2. From the Tools menu, click **Download**. The EasyDownload Complete Project dialog box appears.
3. If the HMI is correctly connected to the computer, then the download process will begin. You will notice that a horizontal bar graph is displayed on the computer indicating how much of the project has been downloaded. The HMI will also display several Write Flash

commands that scroll on the screen. If a timeout error is displayed, please review your connection of the HMI to the computer and make sure that the appropriate settings in EasyManager are used.

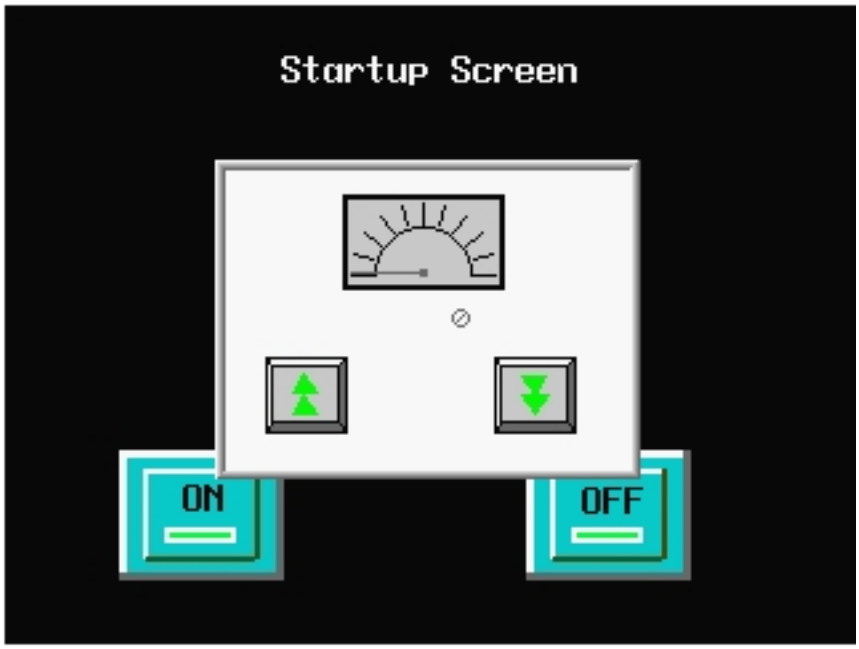
4. After the download is finished, the Mission Complete dialog box is displayed.
5. Click **OK**. The main screen of EasyBuilder reappears.
6. You will now use EasyManager to get the HMI to execute the project you just downloaded. If you want to exit EasyBuilder, from the File menu click **Exit**.

Displaying your project on the HMI

1. From the Windows Task Bar, click **EasyManager**. The EasyManager dialog box appears.
2. Click **Jump To Application**. This will send a command to the HMI to run the project.
3. The HMI should display the following screen.



4. Press the **ON** function key to display the popup window.



5. Press the increment and decrement keys on the HMI to see the value in the data register and scale meter change. Press the **OFF** function key to remove the popup window.

CONGRATULATIONS! You have completed your first EZware-500 project.

Chapter 3 - Simulator Mode

As you saw from creating the sample project in the last chapter, downloading any changes you make to the HMI can take time. To decrease the amount of time required to download a project to the HMI you can make these changes in EasyManager:

- Select the **115200** baud rate. The default setting of 38400 is recommended for older computers or when using a very long programming cable.
- Select **Partial Download/Upload**. The default setting is Complete Download/Upload. A complete download should be used whenever you receive a later version of EZware-500 or any Boot ROM updates. It should also be used whenever you select a different PLC protocol driver.

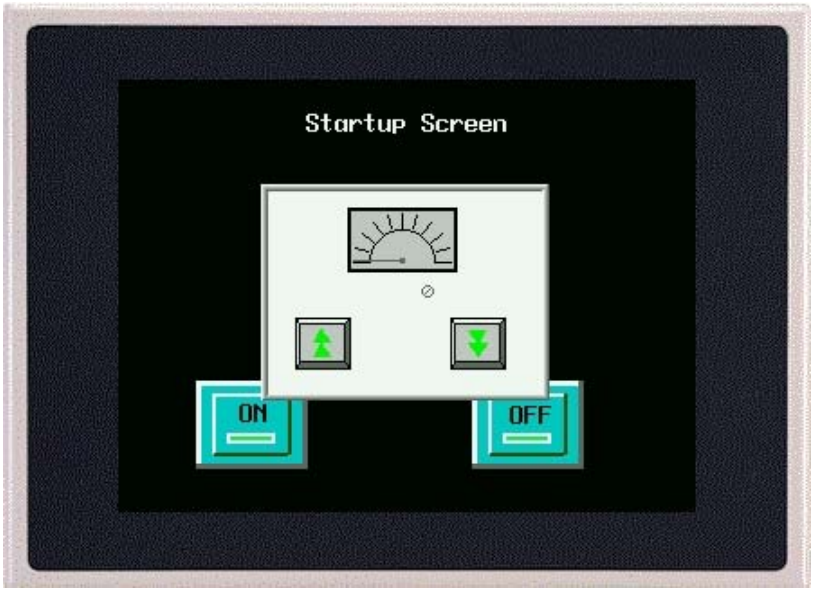
However, even with these improvements, it may still take 1-2 minutes to download a project. A better way of testing changes made to a project is to use the computer to simulate the operation of the HMI.

This chapter shows you how to dramatically save time when creating and testing a project by putting the computer into Simulation Mode. In Simulation Mode, the computer displays a screen that simulates what is seen when the HMI executes the project file. You can use the mouse on your computer to simulate the operation of the touch screen on the HMI. Rather than waiting two minutes to test each change you make to your project, you can now instantly switch to Simulation Mode to see if your project is working as expected.

You can enter Simulation Mode in three ways. Using the sample project that you created in the last chapter, we will guide you through the three methods.

The Simulation Screen

All modes of simulation use the same screen.



For our example, we are showing the sample project as it would appear after the ON function key has been pressed. hen the Simulation Mode screen first appears, it will always show the startup screen of the project. You can test any objects you have created which require touch screen input by clicking on them. In this example, clicking on the increment key simulates the key as it is pressed on the touch screen. Simulation Mode always uses the compiled version of the project that you have created (the *.eob file). Therefore, you must save and compile any changes you make to the project before you can simulate it.

Simulating the HMI is done in on-line or off-line mode. On-line simulation has two modes:

'Normal' mode requires that the HMI be connected to the computer and that the PLC be connected to the HMI.

'Direct' mode requires only that the PC be connected to the PLC. Please note the following:

- Weintek' standard communication cables will not work in Direct mode. The 'HMI' end of the cable must be wired for the PC's serial port.
- To simulate protocols using RS422/485, an RS422/485 card must be installed in the PC.
- The Allen Bradley DH485 and Telemecanique UniTelWay protocols are not supported in Direct mode.

The On-Line simulation mode is selected in EasyManager, by checking or unchecking the Direct Online-Simulator box. Whether started from EasyBuilder or EasyManager, the Online-Simulator will operate in the mode selected by the Direct Online-Simulator box.

On-line simulation is most useful when a project is near completion and you need to test the interaction of the project with the PLC. It is also very convenient when making minor changes to a project because these changes can be instantly checked for operation. When 'normal' on-line simulation is used, the computer interrogates the PLC through the HMI to access any data required. When 'direct' on-line simulation is used, the computer interrogates the PLC directly.

Off-line simulation does not require any connection to the HMI. Because of this, off-line simulation is most often used when starting a project. You can quickly test new ideas and create preliminary screens without the HMI. Off-line simulation is also great for demonstrating the operation of the HMI and becoming familiar with the operation of the HMI before it is installed. If you want to simulate the interaction with a PLC when off-line, it can be done by creating additional windows on the HMI which allow PLC data input.

To use off-line simulation mode from EasyBuilder:

1. If the project has been modified since the last time it was saved and compiled, or if it has not yet been compiled, it must be saved and compiled before the simulator can start.
2. From the Tools menu, click Off-line Simulation or click the Off-line Simulation icon in the Standard toolbar.
3. The Simulation Mode screen should appear.
4. To end off-line simulation press the Space bar or right click in the simulation screen and click **Exit**.

To use on-line simulation mode from EasyBuilder

1. If the project has been modified since the last time it was saved and compiled, or if it has not yet been compiled, it must be saved and compiled before the simulator can start.
2. From the Tools menu, click On-line Simulation or click the On-Line Simulation icon in the Standard toolbar.
3. The Simulation Mode screen should appear.
4. To end on-line simulation press the Space bar or right click in the simulation screen and click **Exit**.

Simulation mode can also be started from EasyManager. EasyManager requires only the compiled project file (*.eob). This can be useful in situations in which you want to demonstrate changes made to a project for a client without giving him access to your project file. It also allows you to rapidly switch from simulating one project to another without having to open, save, and compile each project.

To use off-line simulation mode from EasyManager

1. Start the EasyManager software.
2. Click **Offline-Simulator**. The Open Project box appears.

3. Click on the compiled project file that you wish to simulate.
4. Click **Open**. The simulation screen should appear.
5. To end off-line simulation press the Space bar or right click in the simulation screen and click **Exit**.

▶ **To use on-line simulation mode from EasyManager**

1. Start the EasyManager software.
2. If using Direct Online Simulation, check the **Direct Online-Simulator** box. If using Normal Online Simulation, leave unchecked.
3. Click the **Online-Simulator**. The Open Project dialog box appears.
4. Click on the compiled project file that you wish to simulate.
5. Click **Open**. The simulation screen should appear.
5. To end on-line simulation, press the Space bar or right click in the simulation screen and click **Exit**.

Troubleshooting Tools During Simulation

Simulation mode allows additional functionality. By placing the mouse anywhere on the simulation screen and doing a right click, a new menu comes up with additional features.

Search: Allows searching the entire project for a particular PLC address. To search, check the **PLC** and then type the *address.* Searching can also be done by parts. Check **Part** and select the type of object.

PLC Monitor: Allows monitoring of the polling communication between the HMI and the PLC. This monitor can help troubleshoot register fault problems. To view each time the HMI polls the PLC and grabs a single or a block of registers, click on the **Block Capture**. Each poll communicates the following information:

- The window from which the register or block of registers is called (Win000 are background tasks such as alarm scan objects)
- Time it took to get the data
- Register address
- Number of 16-bit words it grabbed
- Register type

If Block Capture fails to retrieve the data, the message “FAIL” will appear to the right.

System Resource: Displays information about the availability of the following HMI resources:

- RAM memory
- Timers
- Queues

Data Monitor: Shows live PLC data polling based on individual object types.

Emulator Setting: Sets up how the Simulator appears and operates.

Print Screen (preview & save to file): Prints a picture of the current screen displayed. To preview before printing, select Screen Preview. “Print screen to a file” will create a bitmap image of the screen.

Print Window (preview & save to file): To print object/address information within, select **Window printing** and then select the **Table** radio button. To preview, select **Window Preview**. Selecting **Print window to a file** creates a bitmap image of the screen data.

Capturing Simulation Screens to Use as Documentation

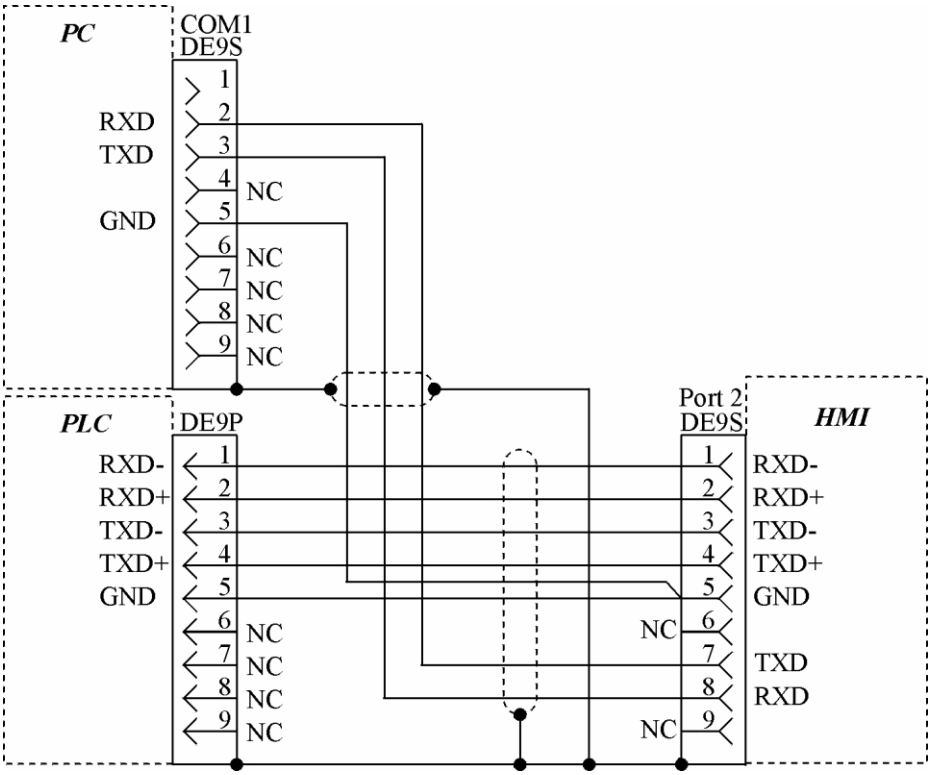
▶ **To capture screen pictures for creating documentation**

1. Run the simulator (either on-line or off-line)

2. When the window to be captured is displayed, hold the ALT key down and push the PRINT SCREEN key on your keyboard. This will send a bitmap image of the simulation window to the clipboard.
3. Open a word document or your publishing software and PASTE (CTRL + V) the image from the clipboard.

Wiring for Normal Simulation Mode

This section describes how to connect the PC/PLC/HMI for Online Simulation in the "Normal" mode. To configure ıe HMI using Simulation Mode, you will need an HMI Configuration Cable, Maple P/N 7431-0098. Connect the end ırked "HMI" into Port 2 on the HMI, marked "PLC[RS-485]/ PC[RS-232]," and connect the end marked PC into the ɔper COM port on your PC. Plug the end marked "PLC" into the appropriate port on your PLC. See the figure below for serial port pin assignments and the following two figures on the next page for connecting the Silver Series to a PC and a PLC.



HMI to PC Serial Port Pin Assignments

The following section describes how to connect the PC/HMI for Online-Simulation in the 'Direct' mode.

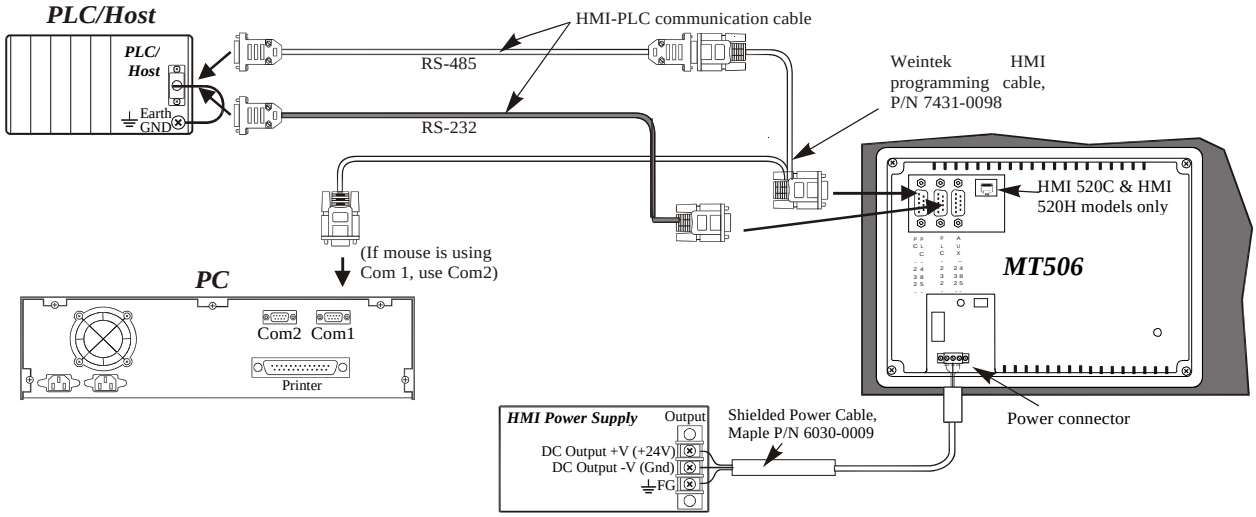
Note the following restrictions:

- Weintek' standard communication cables will not work in Direct mode. The 'HMI' end of the cable must be wired for the PC's serial port.
- To simulate protocols using RS422/485, an RS422/485 card must be installed in the PC.
- The Allen Bradley DH485 and Telemecanique UniTelWay protocols are not supported in Direct mode.

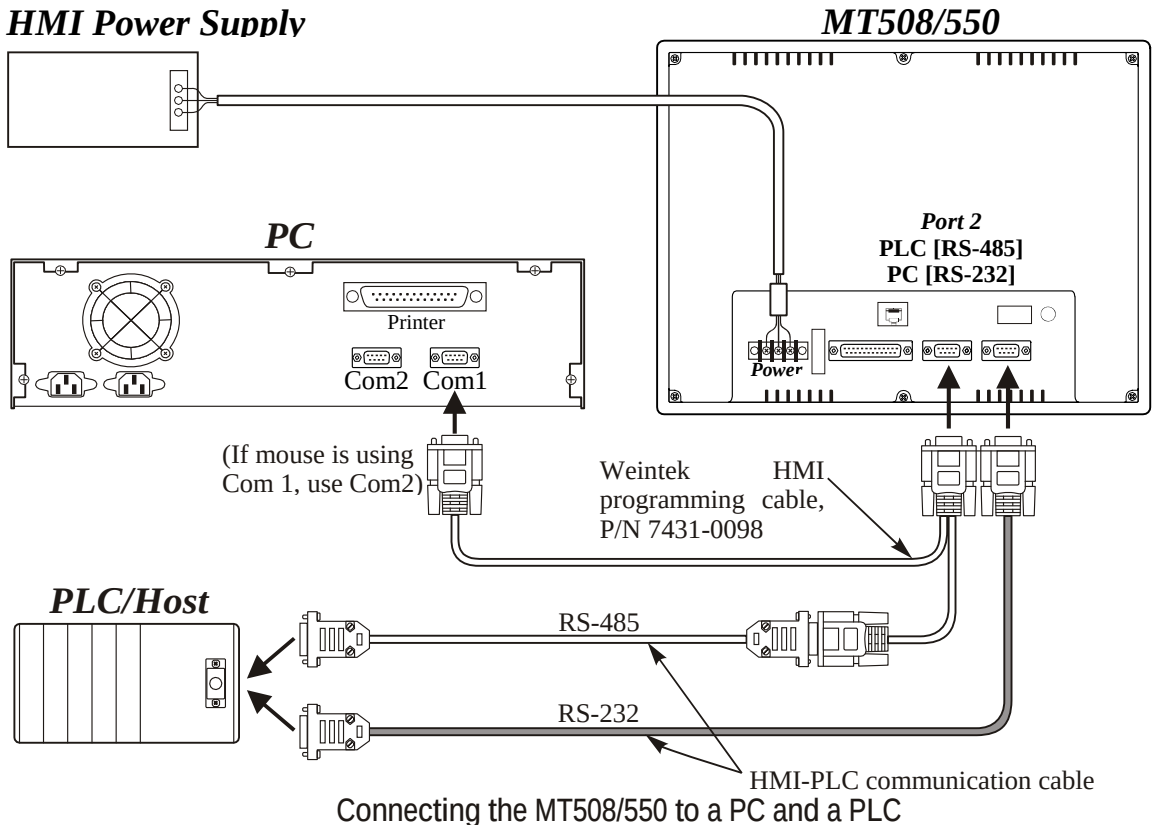
To configure Direct Online-Simulation, you will need a cable connecting the PC to the PLC. Standard Maple communication cables will not work. The cable needs to be wired such that the PLC's RX pin(s) is connected to the

PC's TX pin(s), and the PLC's TX pin(s) is connected to the PC's RX pin(s). Any common pins also need to be brought through, and shield connections should also be made.

Refer to Weintek communications cable drawings for guidance, but remember that the HMI end will have to be changed. Refer to the figures on the previous pages for information on power connections and HMI serial connector pinouts.



Connecting the MT506 to a PC and a PLC



Connecting the MT508/550 to a PC and a PLC

Now that you are familiar with using simulation mode, the next chapter guides you through the fundamental operation of three primary segments of the EZware-500 configuration software.

Chapter 4 - Using EZware-500

Overview

The EZware-500 software is composed of three separate applications which are accessible from the EZware-500 folder: EasyBuilder, EasyManager, and EasyASCIIFontMaker. EasyBuilder is the application software used to create a project file. EasyManager is a utility application that puts the HMI into different operating modes. Finally, the EasyASCIIFontMaker program allows you to modify the text fonts that are used in EasyBuilder.

By the end of this chapter, you should be quite familiar with the operation of EasyManager and EasyASCIIFontMaker. You will also be able to maneuver around EasyBuilder easily. This will pave the way for actually creating graphics objects in later chapters.

The EasyManager

The EasyManager utility sets the communications parameters that are used by the computer when communicating to the HMI. The utility can also be used to change the operating mode of the HMI.
The following illustration shows the various commands available:

Communications Settings

- **Com Port** Select COM1 – COM10 on the computer to download a project to the HMI.
- **Baud Rate** Select 38400 or 115200 bps to download or upload a project to the HMI. We recommend that you use 115200 unless your computer is unable to transmit at that speed.
- **Project or Recipe Download/Upload** Select Project Download/Upload when you want to download the project. Select Recipe Download/Upload to update your recipe files (recipe module only available for some models).
- **Complete or Partial Download/Upload** When you first program the HMI you should select a complete download. A complete download should also be used whenever a new version of EZware-500 is used or you receive a Boot ROM update. Finally, a complete download should be performed whenever the PLC protocol driver is changed. If, however, you are making changes to

Operating Modes

- Click **EasyBuilder** to start the EasyBuilder application software.
- Click **Online-Simulator** to open a compiled project file (*.eob) in on-line simulator mode. Check Direct Online-Simulator to perform online simulation without using the HMI.
- Click **Offline-Simulator** to open a compiled project file (*.eob) in off-line simulator mode. See the chapter “Simulator Mode” for more information.
- Click **Download** to send a compiled project file stored on the computer to the HMI.
- Click **Upload** to receive a compiled project file from the HMI to be stored in the computer.

*Files uploaded from the HMI are “.eob” files. These need to be decompiled, before they can be modified by the EasyBuilder software. To decompile, go to the **Tools** menu, and select **Decompile**. Browse for your file and then press the “Decompile” button. This will create a “.epj” file, which can be opened and modified in EasyBuilder. (If the project was created in a version of EasyBuilder prior to v.2.0, it will not decompile.)*

- Click **Jump To RDS** to place the HMI in Remote, Debug, Simulation mode. This mode is used only when you need to update the Boot ROM.
- Click **Jump To Application** to place the HMI in normal application mode. This command must be sent to the HMI to get the HMI to execute the project file that is stored inside of it.
- Click **Jump To Touch Adjust** to calibrate your touch screen. The HMI displays a crosshair cursor and you are asked to touch the crosshair cursor for:
  - the top left position
  - the top right position
  - the bottom left position
  - the bottom right position
- Two rectangle objects then appear on the HMI screen. To determine if the touch screen is properly calibrated, place and hold your finger somewhere on the touch screen other than on the rectangle spots. The crosshair cursor should appear directly beneath your finger. If necessary, press the left rectangle to repeat the calibration. Press the right rectangle to end calibration.
- Once calibration is complete, press the touch screen again to go back to normal application mode.

The touch screen is always calibrated at the factory, so it shouldn’t require calibration under normal circumstances.

That's all there is to EasyManager! Most of the time, you will use EasyManager to start the project that you have downloaded from EasyBuilder to the HMI by clicking the **Jump To Application** button.

The EasyASCIIFontMaker

The EasyASCIIFontMaker application allows you to modify the text font characters that are supplied with EZware-500. This feature of the MT5xx HMIs enables you to create any special text characters that you may need In fact, you can even create an entirely new font!

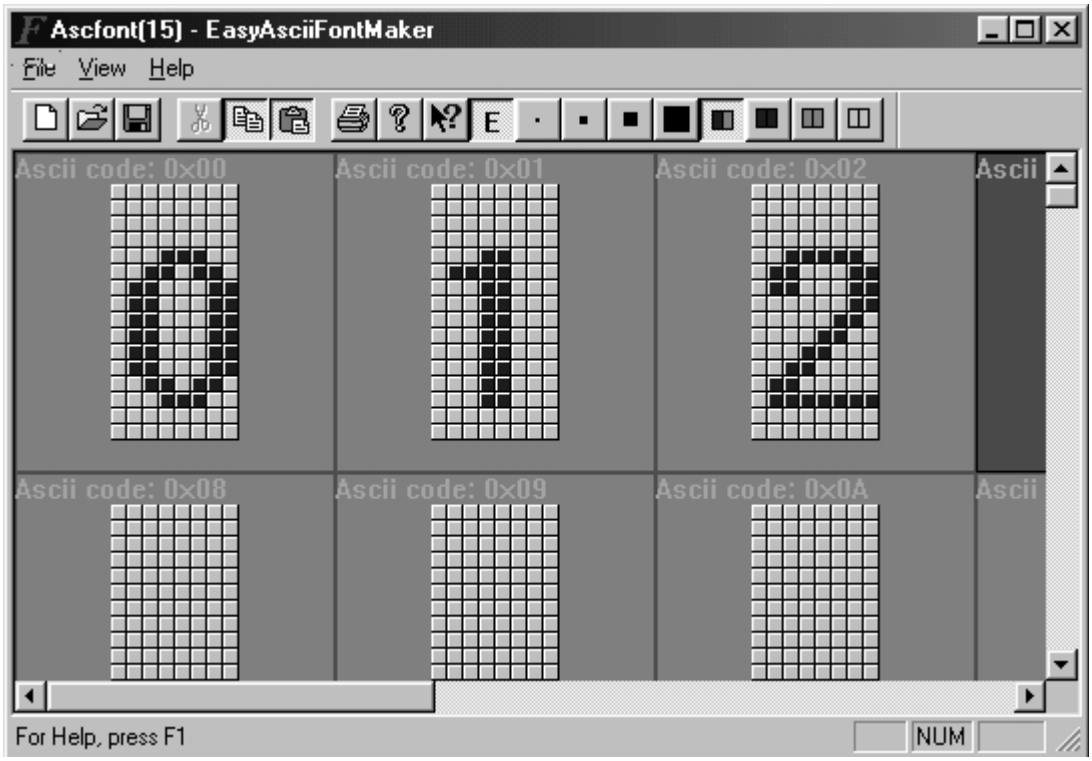
EasyBuilder's Default Text Fonts

EasyBuilder uses three font types to display text in eight different font sizes: 8, 16, 24, 32, 48, 64, 72, and 96.

- The **Type 8** font is used to display the size 8 font characters. The font characters are stored in a file in the EZware-500 directory called ASCFONT.8.
- The **Type 16** font is used to display the size 16, 32, and 64 font characters. These font characters are stored in a file called ASCFONT.16.
- The **Type 24** font is used to display the size 24, 48, 72, and 96 font characters. These font characters are stored in a file called ASCFONT.24.

Using the EasyASCIIFontMaker

The following illustration shows the ASCIIFontMaker application screen with the ASCFONT.16 file opened:



To modify a font file

1. Start the EasyASCIIFontMaker application.

You must close the EasyBuilder application before you can modify a font file.

2. Click **Open** from the File menu. Open the font file you wish to modify.
3. Click **Edit** from the View menu to place the application into edit mode.
4. Click **Toggle** from the View menu to be able to modify a character.
5. Click the particular character that you wish to modify.
6. Click on any square with the character block to draw. Clicking the same square again while in Toggle mode will clear the square.
7. Right-click to Copy, Paste, or Clear a character.
8. After you are satisfied with the changes made, click **Save** from the File menu.

It is generally good practice to copy the font files to backup files before you make any changes so that you can restore them later, if necessary.

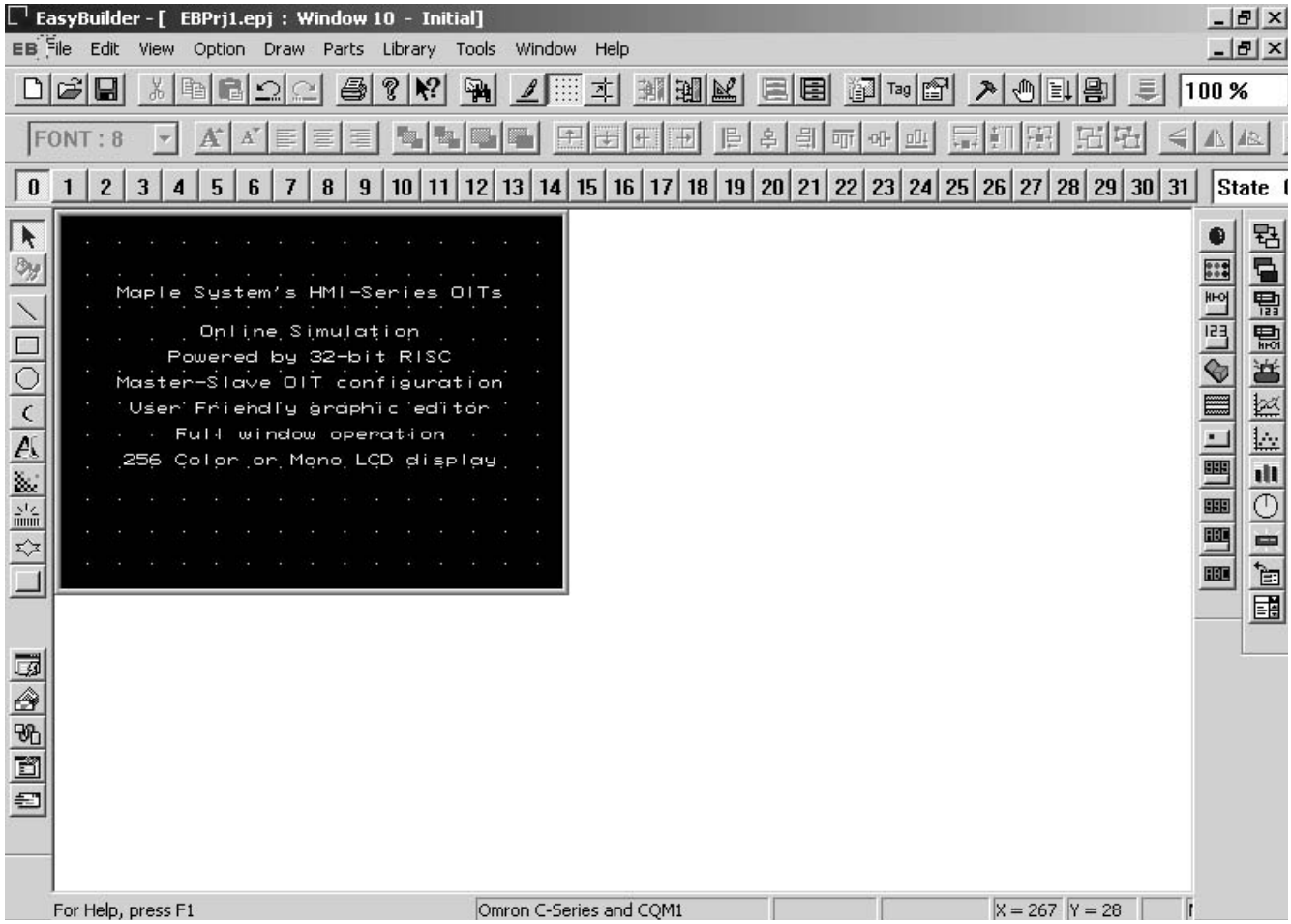
The table below describes the use for each icon on the toolbar.

Icon	Name	Description
	New	Starts a new font file.
	Open	Opens an existing font file.
	Save	Saves the font file.
	Cut	Not currently used. Right click on any character to copy, paste, or clear it.
	Copy	Not currently used. Right click on any character to copy, paste, or clear it.
	Paste	Not currently used. Right click on any character to copy, paste, or clear it.
	Print	Not currently available.
	About	Used to show the version of EasyASCIIFontMaker.
	Help	Not currently available.
	Edit	Puts the application into Edit mode so that characters can be modified.
	View x1	Shows how the characters look in various Zoom modes.
	View x2	Shows how the characters look in various Zoom modes.
	View x3	Shows how the characters look in various Zoom modes.
	View x4	Shows how the characters look in various Zoom modes.
	Toggle	In Edit mode, using this format allows you to click once to fill a square in the character block. Click again to clear the square.
	On	In Edit mode, this format always fills a square when it is clicked.
	Off	In Edit mode, this format always clears a square when it is clicked.
	None	Use this format to ensure that no characters are changed.

The EasyBuilder Application

This section guides you in how to operate the EasyBuilder application. However, it does not show you how to program your HMI or how to create graphics objects. These topics are reserved for later chapters. This section shows the fundamental operation of EasyBuilder from saving files, printing projects, and selecting the target PLC to showing how graphics objects can be easily manipulated in the EasyBuilder work area. When you have completed this chapter, you will be better able to use the features that are explained in later chapters.

The following illustration is used for reference to the following sections:



Managing Projects

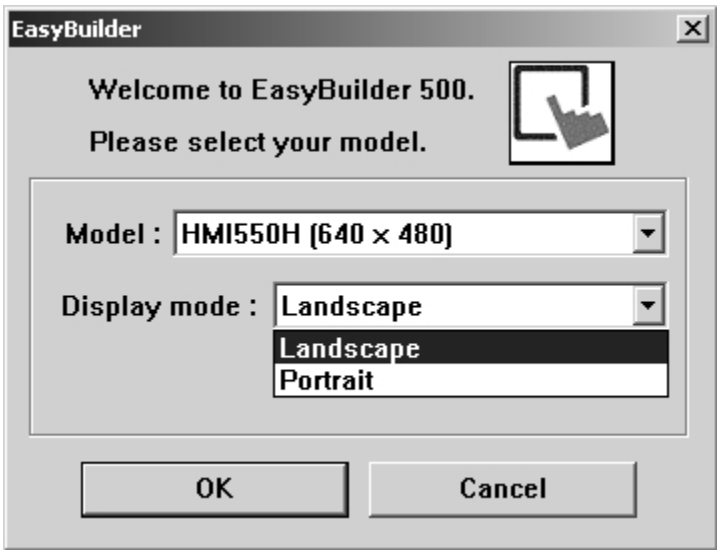
Like most Windows™ application software, EasyBuilder will open, save, close, and print files using the standard windows format.

Opening, Editing Projects

To create a new project 

1. On the **File** Menu, click **New** or click the **New** icon in the Standard toolbar. The EasyBuilder dialog box appears.
2. Select the HMI model you intend to use with your project.

3. Select the Display Mode you intend to use with your project.



4. Landscape mode is the standard display mode for conventionally installed HMIs, with 640 x 480 or 320 x 240 resolution. Portrait mode configures display on HMIs installed at right angles, with resolution of 480 x 640 or 240 x 320.

Landscape

Portrait

A project's display mode may be changed after the initial selection has been made. To set Display Mode select ***Edit-System Parameters…****, and select the* ***Hardware*** *tab. To change the Display Mode, resize the window's resolution to fit the mode selected.*

5. Click **OK**. The main screen of EasyBuilder appears with a blank work area.

To open an existing project 

1. On the **File** Menu, click **Open** or click the **Open** icon in the Standard toolbar. The Open dialog box appears.
2. Click on the project file you intend to open.
3. Click **Open**. The main screen of EasyBuilder appears with the initial screen of the project displayed.

To close a project

1. On the **File** Menu, click **Close**. If changes have been made to the project file, EasyBuilder will ask you if you would like to save the project. Then the main screen of EasyBuilder will remain but with no work area displayed. You must now use the Open or New commands to edit a project.

To save an existing project 

1. On the **File** Menu, click **Save** or click the **Save** icon in the Standard toolbar.
2. If the project already has a name, then the project will automatically be saved. If this is a new project, then the **Save As** dialog box appears.
3. Enter a file name and then click **Save**.
4. The main screen of EasyBuilder reappears.

To save a project using the compress feature

The compress feature allows you to save a project in compressed format so that the project data takes less space on your hard drive. This utility also will save the graphics libraries associated with the project into the project file. This facilitates sending a copy of the project to another person who has EZware in order to download the project into an HMI.

1. From the **Tools** menu, select **Compress/Uncompress...**. The Compressing dialog box appears.
2. In the **Compress** frame box under **Source Name**, select the project (*.epj) file that you wish to compress. File must have a .cmp extension.
3. Click **Compressing…** to begin. The utility will compress the project file and all related graphics libraries into a single file.

To extract a compressed project file

1. From the **Tools** menu, select **Compress/Uncompress...**. The Compressing dialog box appears.
2. In the **Uncompress** frame box under **Source Name**, select the project (*.cmp) file that you wish to uncompress.
3. Click **Uncompressing…** to begin. The utility will extract the project file and all graphics libraries from the *.cmp file.

To exit EasyBuilder

On the **File** Menu, click **Exit** or click on the standard windows **Close** icon in the upper right corner.

Printing Projects

A text-based project summary can be printed from within EasyBuilder. Printing from the Project Simulator allows more options, including generic pictures of the screens. To print the summary, select **Print Object Summary** from the **File** menu. Other options include **Print Preview** and **Print Setup**.

Printing project data is done from the Project Simulator

1. Start the project in the On- or Off-line simulator.
2. Display the window to be printed.
3. Right-click anywhere in the window.
4. To print or save a copy of the window, select Print Screen or Print Screen to File. To print object/address information for the screen, select Print Window or Print Window to File. A list of active windows is displayed. Select the desired window. Select Image to display an image of the window, or Table to display object/address information.

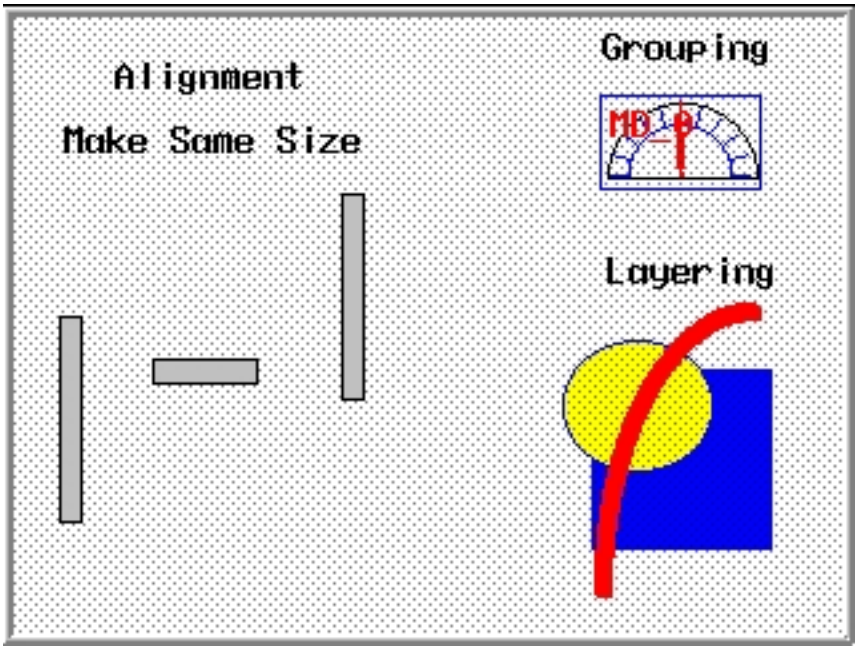
Editing and Creating Screen Objects

This section shows how to manipulate graphics objects that are placed onto the work area of EasyBuilder. We will use examples from sample projects that are included with the EasyBuilder software: MT506M.EPJ, MT506C.EPJ, MT508C.EPJ, and MT510H.EPJ. Please load one of these project files and have EasyBuilder ready before you begin this section.

We will refer to Window 12 of the project. To display Window 12 on EasyBuilder, perform the following steps.

To open a window

1. On the **Window** Menu, click **Open Window**. The Open Window dialog box appears.
2. Click **Window 12**.
3. Click **Open**. Window 12 appears in the work area of EasyBuilder.



Display Options

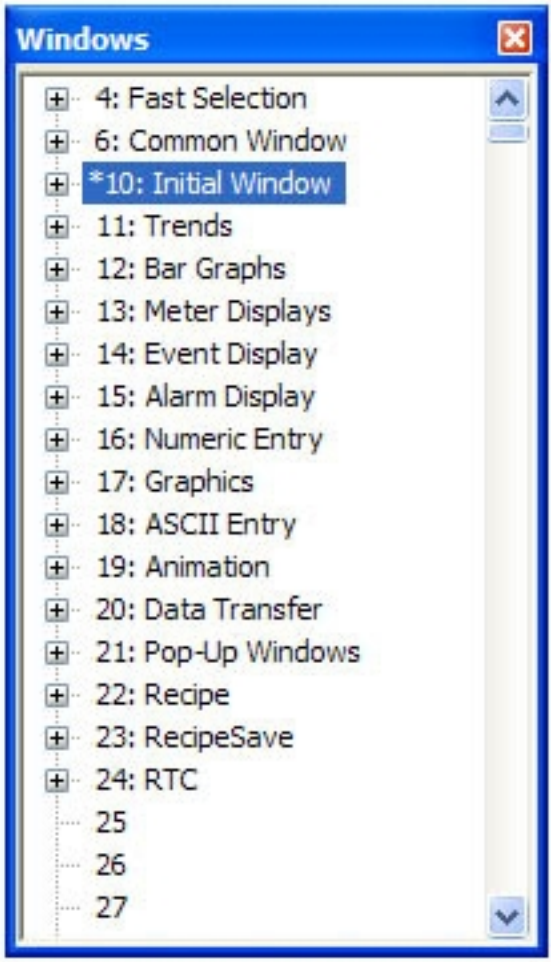
Before we begin describing some of the commands that can be used to edit graphics objects, there are a few commands that apply to the general work area of EasyBuilder.
EasyBuilder provides a **Window No. Treebar** window that can be used to easily maneuver between windows of a project. It also allows you to select any object that is on the window that is currently displayed.

To display/hide the Window No. Treebar

1. On the **View** Menu, click **Window No. Treebar**.

Using the Window Treebar to display window screens

1. The Window Treebar combines into a single tree a list of all of the windows in a project, and all of the objects on each window. Using the 520CDEMO.EPJ project, the following should be seen :

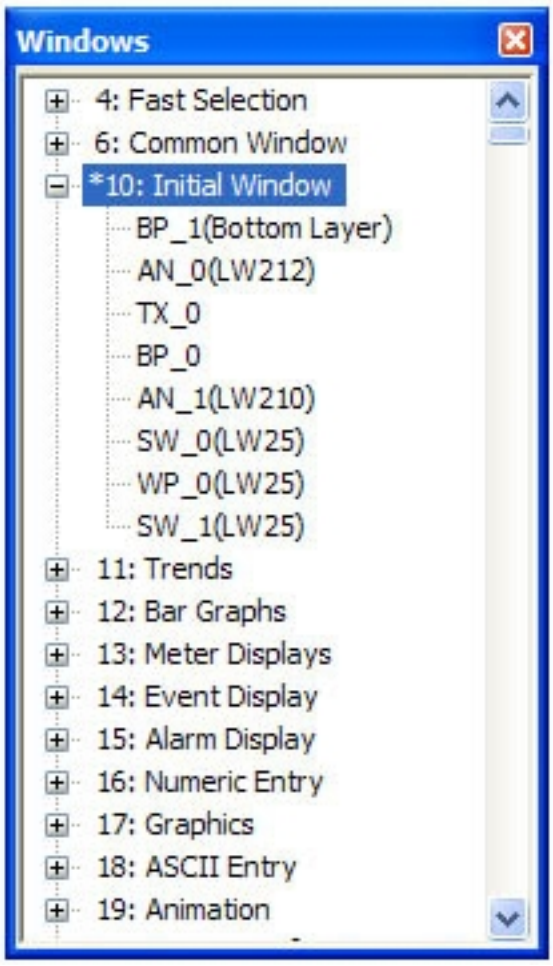


2. Notice that the tree displays the title for each window next to the window number. This makes it easy to determine windows have been created for a project. The asterisk next to Window #10 indicates that this window is currently open.
3. To display an open window, click on the *window number* in the Window Treebar. To display a window that is not yet open, double-click on the *window number*. This will automatically open the window and display it.

4. To change any settings or close an open window, click on the *window number* to select it. Then right-click anywhere inside the Window Treebar to display a pop-up dialog box. Select **Close** to close the window or **Setting** to change any of the window settings.
5. Finally, to create a new window click on any window number that is not currently used. Then right-click inside the Window Treebar and click **Create**.

Using the Window Treebar to change object attributes

1. On the Window Treebar, click + adjacent to any window number. The tree will expand to list the objects on that window, as shown:



2. The Object ID of each objects will be listed. For example, TX 0 is Text Display #0, FK 0 is Function Key #0, etc.
3. To highlight a particular object on the screen, click on the Object ID in the list. This allows the quick selection of a particular object, especially on crowded screens or screens where objects may be overlapping..
4. To display the object's attribute dialog box, double-click the *Object ID* in the Window Treebar.

Using the grid function 

One option is to have the work area covered with grid lines. These grid markings can be helpful when trying to align objects that are created on the work area. If grid lines are selected, you can decide what size each grid is.

1. On the **Option** Menu, click **Grid/Snap**, then click the **Display** box or click the **Grid** icon in the Standard toolbar.

If you still don't see the grid after you have enabled it, make sure that the color of the grid doesn't match the background color of the window currently displayed.

Changing the grid size

1. On the **Option** Menu, click **Grid/Snap**.
2. Select the *X spacing* and *Y spacing* for grid size.

Spacing is measured in pixels.

Changing the grid color

1. On the **Option** Menu, click **Window Property**. The Window Property dialog box appears.
2. Click the pull down box from the Color box. The Color dialog box appears.
3. Click on the appropriate *color box*, then click **OK**. The color box should reflect what you have chosen.
4. Click **OK**.

Another option is to display what are known as Object ID tags. Every time you create a new object in EasyBuilder, an Object ID tag is assigned to the new object. This is done for several reasons:

- Object ID tags are required by EasyBuilder to differentiate each object created.
- If an error occurs during the compile process, EasyBuilder can refer to the window and object ID to indicate the object that is causing problems.

To enable/disable Object ID tags

1. On the **Option** Menu, click **Window Property**. The Window Property dialog box appears.
2. Click the **Display Object ID** checkbox to display tags.
3. Click **OK**.

If you enable the Grid on the work area, you may also take advantage of a useful feature called Snap. The Snap option causes all objects placed into the work area to fall along the boundaries set by the grid lines. The Snap option can help provide a more ordered appearance to graphics objects.

To use the snap option 

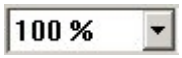
1. The **Grid** option must be enabled.
2. On the **Option** Menu, click **Grid/Snap**. The Grid/Snap Settings dialog box appears.
3. Click the **Snap** checkbox.
4. Click **OK**.

Finally, you will find that you can move objects by selecting them with the mouse cursor and dragging them to a new location. However, you may find it difficult to select an object *without* accidentally moving it just a little. At times this may be frustrating so you may want to disable the move feature using the mouse and only move an object

by changing it's X and Y position parameters in the Profile tab of the object's attributes box. This is the intent of the **Fix Objects** feature.

To enable the fix objects command

1. On the **Option** Menu, click **Grid/Snap**. The Grid/Snap Settings dialog box appears.
2. Click the **Fix Objects** checkbox.
3. Click **OK**. Movement by mouse cursor is now disabled.

To zoom in on the selected window 

Use the zoom pulldown menu at the top right of the screen.

To change the state of the selected window 

1. Use the zoom pulldown menu at the top right of the screen. The objects will appear as they would in the selected state.

To change the language to enable language specific characters

1. On the **Option** Menu, click **Language**. Then select your appropriate option. To use a particular language, EasyBuilder must be installed in that version of Windows.

Basic Editing Commands

To select a graphics object 

1. On the **Edit** Menu, click **Select** or click the mouse cursor icon in the Draw toolbar.
2. Click on the graphics object. For example, using Window_12, click on **Alignment**. This causes the text box to be selected, with small square blocks around the edges indicating the boundaries of the object. Deselect the object by clicking somewhere else in the work area.
3. When a graphics object is selected, it can then be modified, copied, deleted, or moved to a new location.

To select multiple graphics objects

1. On the **Edit** Menu, click **Select** or click the mouse cursor icon in the Draw toolbar.
2. Click and hold down the left mouse button at the upper left corner of the graphics objects you wish to highlight. For example, using Window_12, click to the left and above of **Alignment**. Multiple objects can be selected by holding down the CTRL key and clicking on the objects

3. Move the mouse cursor to the lower right corner of the highlighted objects. Notice that a rectangle is formed as you do this. In this example, move the mouse to the right and below **Make Same Size**.
4. Release the mouse button. The rectangle outline changes to small clear square blocks around the perimeter of the objects selected.

If you do not completely enclose any graphics objects you wish to highlight then they will not be selected.

5. When several graphics objects are selected, they can easily be moved, deleted or copied together.

To select next object 

1. This option can be used to easily select an object that is underneath another object on the screen. To use this option, you must first select the overlapping object.
2. On the **Edit** Menu, click **Select Next Object** or click on the **Select Next Object** icon on the Standard toolbar. You can also right-click on the top-most object on the screen. This will display a menu with a list of the overlapping objects on the bottom. Then check the object that you wish to access.
3. Once the object you are trying to access is selected, you can proceed to modify its attributes or delete it.

To select all objects

1. On the Edit Menu, click Select All Objects.
2. Small clear square blocks appear around the perimeter of all the objects on the window.
3. You can now easily move, delete or copy the entire window screen to a new window.

Using the Undo  and Redo  commands

1. The Undo command is used to cancel the last command or action that you made. For example, select the **Alignment** text box in Window_12.
2. Press the Delete key on your keyboard to delete the text box.
3. From the **Edit** menu, click **Undo** or click the Undo icon from the Standard toolbar. You can also press CTRL+Z. The deleted text box reappears.
4. The Redo command is used to cancel the Undo command. For example, if you decided that you really did want the **Alignment** text box deleted, click the **Redo** command.

Using the Cut  , Copy  , and Paste  commands

1. These commands are all selected from the **Edit** menu or by clicking the appropriate icon in the Standard toolbar.
2. Select the graphics object or objects you wish to cut or copy.
3. Click **Cut** to copy and remove the graphics object(s) from the work area or click **Copy** to copy the graphics object(s). Using Window_12, select the scale meter and then press CTRL+X to cut the object from the work area.
4. Objects cut or copied from one window can be pasted into other windows. Once the object has been selected and cut or copied, open another window, then paste the object into that window.
5. In this example, paste the scale meter back into Window_12 by pressing CTRL+V. The pasted object appears highlighted in the upper leftmost corner of the work area.
6. Move the scale meter by clicking on the highlighted object and dragging it back to the original position.

Using the Multi-Copy command

The Multi-Copy command is used to make multiple copies of a single object on a screen. This is most useful when creating new keyboards or data tables. In addition to copying the object, this command will automatically assign new PLC memory addresses to each object, (applies only to Bit-type objects).

1. Select the object you wish to copy.
2. From the **Edi**t menu, select **Multi. Copy...**. The Multi. Copy dialog box appears.
3. Modify the settings according to your requirements.
4. Click **OK**. The main screen of EasyBuilder is redisplayed with the Multi. Copy command executed.

Multiple objects can be created according to these parameters:

- **Pitch vs. Interval:** This setting affects how EasyBuilder places the copies of an object on the screen relative to each other. The Pitch setting will interpret the X Distance setting as the distance from the left side of an object to the left side of the next object and the Y Distance setting as the distance from the top of one object to the top of another object.

  The Interval setting will interpret the X Distance setting as the distance from the right side of one object to the left side of the next object. The Y Distance setting is the distance from the bottom of one object to the top of the next object.

- **RGT vs. BTM:** This setting determines in what order each new copy is made. This will affect how the automatic PLC memory addressing is created for each object. The RGT (right) option creates copies along the X axis first (from left to right), then along the Y axis (from top to bottom). The BTM (bottom) option creates copies along the Y axis first, then the X axis.

- **X Distance, Y Distance:** These settings determine the spacing (in pixels) between objects.

- **Quantity X, Quantity Y:** These settings determine the number of copies to be made along the X axis (horizontal) and the Y axis (vertical).

- **Adjust Distance:** This is the offset used when assigning a new PLC memory coil address to each new copy. For example, if 1 is selected, then EasyBuilder will add one to the PLC memory coil address for each new copy made.

Using the Window Copy command

The Window Copy command is used to copy a window from an existing project into the project that you are currently editing. This feature reduces time spent creating new projects since you can now use windows that you created for other similar projects.

1. From the **Edit** menu, select **Window Copy...**. The Window Copy dialog box appears.

2. In the **Source Project:** box, specify the path to the project that contains the window screen you wish to copy. Use the **Browse** command button if you are unsure of the location or name of the project file.
3. In the **Source Window No.:** box, enter the number of the window screen you wish to copy.
4. In the **Desti. Window No.:** box, enter the number of the window screen in the project you are currently editing that you wish to copy the contents of the source window to.
5. Click **OK**. The main screen of EasyBuilder is redisplayed with the Window Copy command executed.

To delete a graphics object(s)

1. Select the object or objects you wish to delete.
2. Press the **Delete** key or from the **Edit** menu, click **Delete**.

You can restore the deleted object by using the Undo command.

To move a graphics object(s)

1. Select the object or objects you wish to move.
2. When you place the mouse cursor over a highlighted object, the cursor changes to a crosshair. This indicates that you are able to move the selected object. Move the object by clicking on it and dragging the object to the proper position on the work area.

To resize a graphics object

1. Select the object to be resized.
2. To resize the object, move the mouse cursor over one of the small black squares. The cursor changes to a double-arrow icon to indicate that it is in resizing mode.
3. Click and drag the mouse to resize the object.
4. Objects can also be resized by changing the width and height attributes in the Profile tab of the Attributes dialog box, (see below).

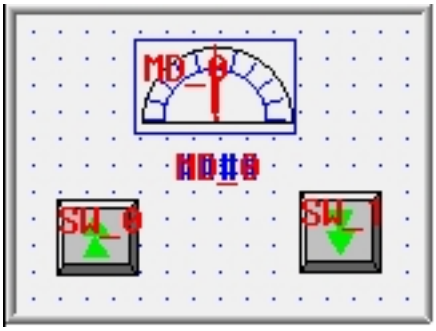
To change attributes of a graphics object

1. Select the object to be changed, then select **Change Attributes** from the **Edit** menu. You can also double-click the object.
2. The object's attribute dialog box is displayed. Object Attributes defines what the object is or how it behaves, (i.e. size, position, color, etc.). Click **OK** to accept any changes made or **Cancel** to cancel any changes.

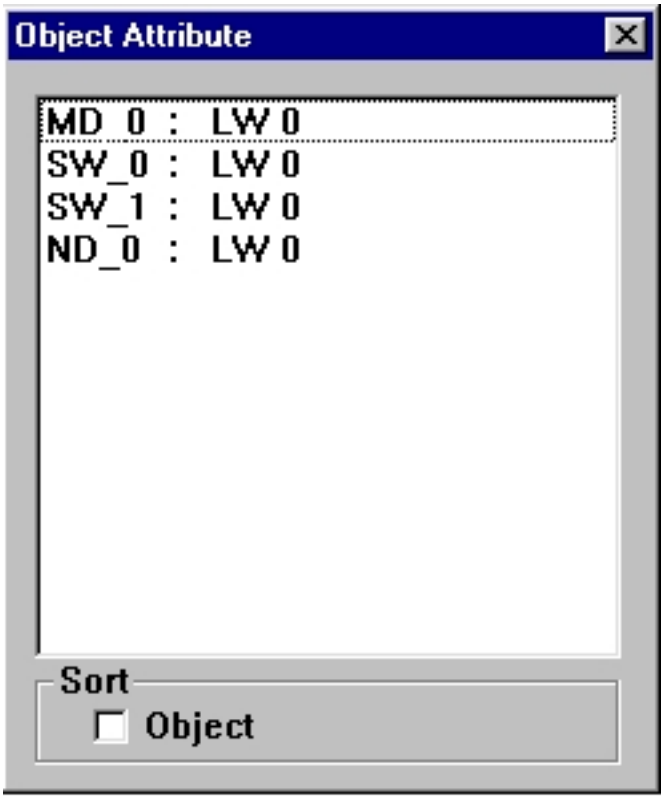
To view object attributes of multiple graphics objects on a window

1. There will be times when you may want to quickly determine what PLC data registers are tied to which objects in a window. This can easily be done using the object attributes command.

2. For this example, open Window 11 to display the popup window of the sample project.



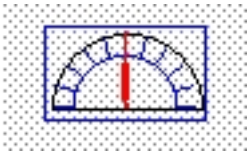
3. On the **Edit** menu, click **Select all objects**.
4. From the **Edit** menu, click **Change Attribute**. The Object Attribute dialog box is displayed.



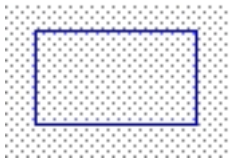
5. All of the objects that use PLC data registers or the internal memory of the HMI are listed in this dialog box according to their Object ID tag. Alongside each ID tag is the PLC or HMI memory identifier.

Grouping Objects

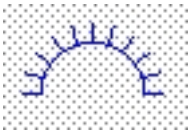
When creating graphics on a window, you may create a complex graphic that is actually composed of several simpler objects overlaid onto each other. The scale meter in Window 12 is such an example.



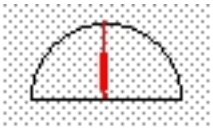
This meter is actually composed of three separate objects:



a rectangle



a scale



a meter display

These objects are then overlaid to represent the scale. *Grouping* the three objects together makes it easier to move or copy the scale meter. Instead of copying each separate object, the scale meter can be assigned as a group, then copied as though it were one object.

To group objects 

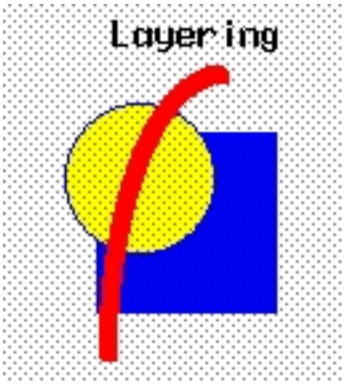
1. Select the objects to be grouped.
2. From the **Edit** menu, click **Group** or click the **Group** icon from the Manager toolbar.
3. All objects within the group can now be copied, deleted, or moved by clicking on the group.

To ungroup objects 

1. Select the objects to be ungrouped.
2. From the **Edit** menu, click **UnGroup** or click the **UnGroup** icon from the Manager toolbar.
3. All objects within the group can now be copied, deleted, or moved separately.

Layering Objects

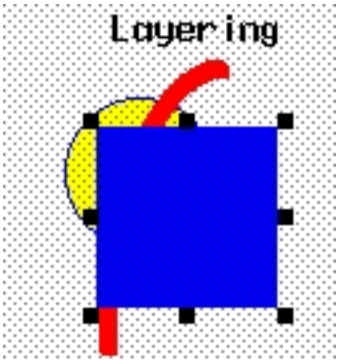
Graphics objects can be overlaid upon each other. When 'layered' the graphics object on the topmost layer will be completely seen. How much of the other graphics objects are seen depends on what is on top of them. The following layer commands help to position the overlaid objects exactly how you want them. To better illustrate how layering works we will use a rectangle object, a circle object and an arc object that are overlaid on each other in Window 12 of the sample project.



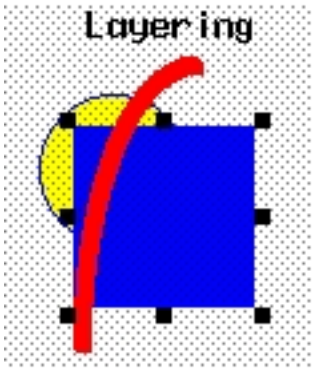
Using the top  **and bottom**  **layer commands**

1. Select one of the layered objects. In this example, select the layered rectangle from Window 12.

2. From the **Edit** menu, click **Layer**, then **TopLayer** or click the **TopLayer** icon from the Manager toolbar. Note that the rectangle completely overlaps the circle and the arc.

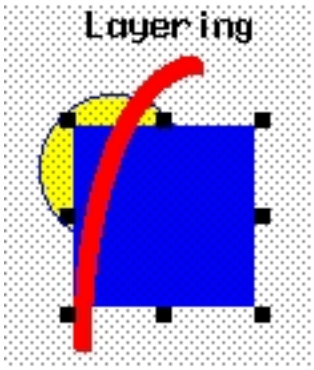


3. From the **Edit** menu, click **Layer**, then **BottomLayer** or click the **BottomLayer** icon from the Manager toolbar. Note that the rectangle recedes behind the circle and the arc.



Using the next  and previous  layer commands

1. These two commands can move an object through the various layers of other objects.
2. Select one of the layered objects. Again, select the layered rectangle from Window_12.
3. From the **Edit** menu, click **Layer**, then **PreviousLayer** or click the **PreviousLayer** icon from the Manager toolbar. Note that the rectangle covers the circle but not the arc.



4. From the **Edit** menu, click **Layer**, then **NextLayer** or click the **NextLayer** icon from the Manager toolbar. Note that the rectangle recedes behind the circle and the arc.

Normally, an object that is controlled by a PLC Register (i.e., a Word Lamp, Bit Lamp, Animation, etc.) is brought to the Top Layer when the value in the PLC Register changes. This behavior can be changed by the Part Layout

option on the Editor tab of the System Parameters dialog. A setting of Control is the default, and sets the behavior as outlined above. A setting of Nature will result in the object remaining at the layer assigned during development.

The Part Layout setting is global, and affects all PLC-controlled objects in the applicati

Nudging Objects

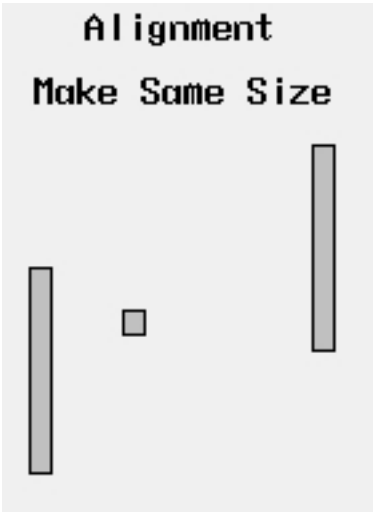
Nudging is used to fine-tune the movement of objects in the work area of EasyBuilder. Using the nudge feature on a selected object will move that object in the specified direction either by one pixel or by the grid setting amount.

▶ **Using the nudge top , bottom , left , and right commands**

1. Select one of the layered objects or a group of objects in the work area of EasyBuilder.
2. From the **Edit** menu, click **Nudge**, then select the direction of the nudge or click the appropriate icon from the Manager toolbar.

Aligning Objects

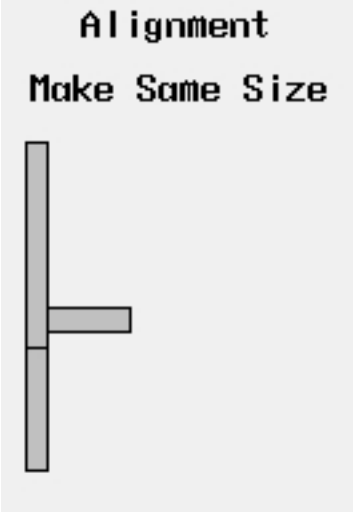
Alignment can be used to quickly align two or more objects. To better illustrate, refer to the left side of Window 12 of the sample project:



▶ **Using the align left command**

1. Select the objects you wish to align. For this example, select the three rectangle objects of Window 12

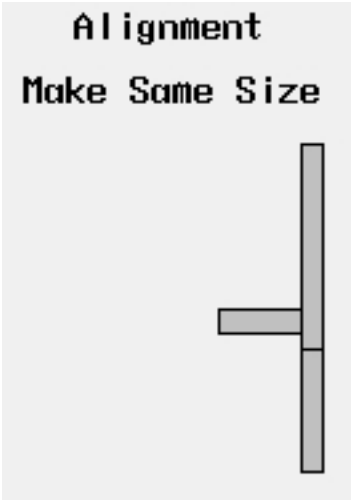
2. From the **Edit** menu, click **Align**, then **Left** or click the appropriate icon from the Manager toolbar.



3. From the **Edit** menu, click **Undo** to put the objects back in their original position.

Using the align right command

1. Select the objects you wish to align. For this example, select the three rectangle objects of Window_12
2. From the **Edit** menu, click **Align**, then **Right** or click the appropriate icon from the Manager toolbar.

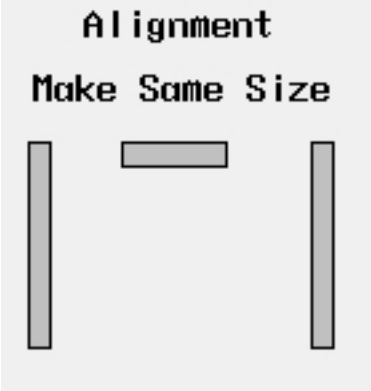


3. From the **Edit** menu, click **Undo** to put the objects back in their original position.

Using the align top command

1. Select the objects you wish to align. For this example, select the three rectangle objects of Window 12

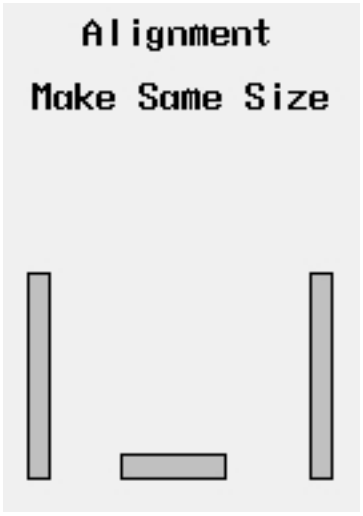
2. From the **Edit** menu, click **Align**, then **Top** or click the appropriate icon from the Manager toolbar.



3. From the **Edit** menu, click **Undo** to put the objects back in their original position.

Using the align bottom command

1. Select the objects you wish to align. For this example, select the three rectangle objects of Window_12
2. From the **Edit** menu, click **Align**, then **Bottom** or click the appropriate icon from the Manager toolbar.

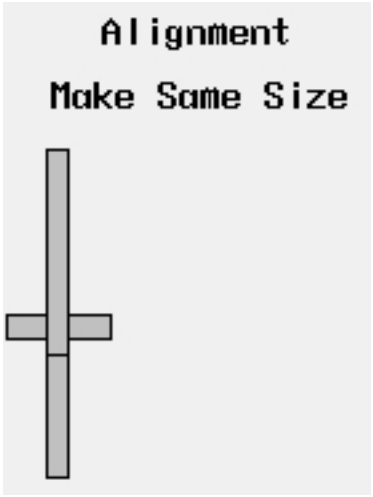


3. From the **Edit** menu, click **Undo** to put the objects back in their original position.

Using the align horizontal center command

1. Select the objects you wish to align. For this example, select the three rectangle objects of Window 12

2. From the **Edit** menu, click **Align**, then **Horiz. Center** or click the appropriate icon from the Manager toolbar.

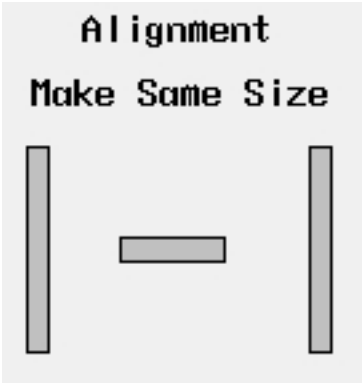


Centering is based on the horizontal center of the left-most object.

3. From the **Edit** menu, click **Undo** to put the objects back in their original position.

Using the align vertical center command

1. Select the objects you wish to align. For this example, select the three rectangle objects of Window_12
2. From the **Edit** menu, click **Align**, then **Vert. Center** or click the appropriate icon from the Manager toolbar.

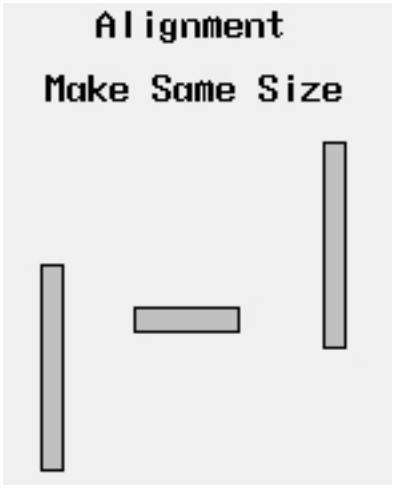


Centering is based on the vertical center of the top-most object.

3. From the **Edit** menu, click **Undo** to put the objects back in their original position.

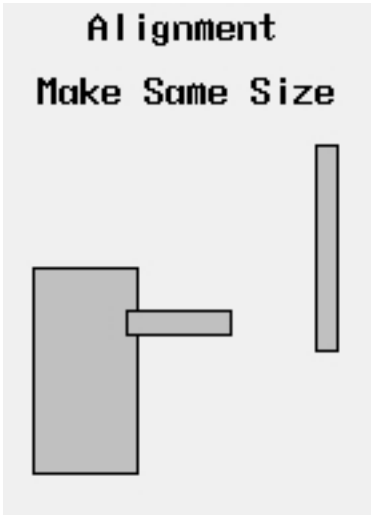
Making Objects the Same Size

is feature is handy if you want to quickly make two or more objects the same size. This is most often used when you are trying to overlap objects that must be the same size. To better illustrate, refer to the left side of Window_12 of the sample project:

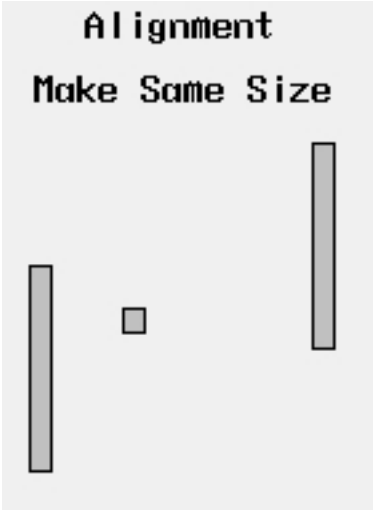


▶ **The make same width**  **command**

1. Select the objects you wish to make the same width. For this example, select the left and center rectangle objects of Window_12
2. From the **Edit** menu, click **Make Same Size**, then **Width** or click the appropriate icon from the Manager toolbar.

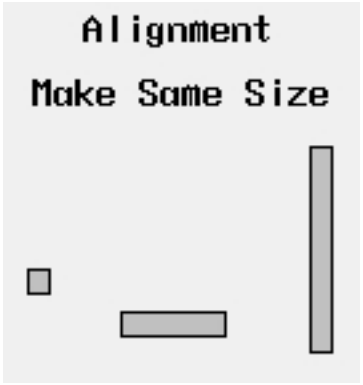


EasyBuilder always uses the width of the last object that was created as the basis for the width. In this example, the left rectangle was created first, then the middle rectangle, finally the right rectangle. See what happens when the ***make same width*** *command is applied to the center and right rectangles.*



The make same height  command

1. Select the objects you wish to make the same height. For this example, select the left and center rectangle objects of Window_12
2. From the **Edit** menu, click **Make Same Size**, then **Height** or click the appropriate icon from the Manager toolbar.

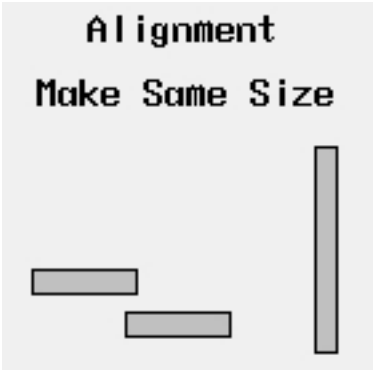


*EasyBuilder always uses the height of the last object that was created as the basis for the height. In this example, the left rectangle was created first, then the middle rectangle, finally the right rectangle. See what happens when the **make same height** command is applied to the center and right rectangles.*

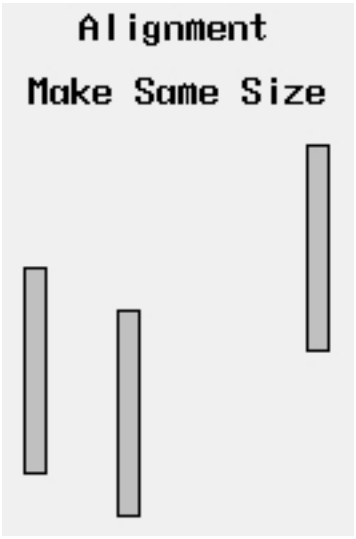
The make same size  command

1. Select the objects you wish to make the same size. For this example, select the left and center rectangle objects of Window 12

2. From the **Edit** menu, click **Make Same Size**, then **Both** or click the appropriate icon from the Manager toolbar.

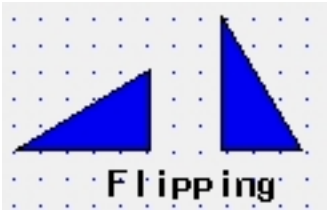


*EasyBuilder always uses the size of the last object that was created as the basis for the size. In this example, the left rectangle was created first, then the middle rectangle, finally the right rectangle. See what happens when the **make same size** command is applied to the center and right rectangles.*



Flipping Objects

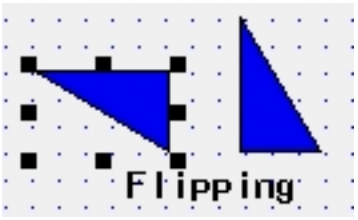
These three commands allow you to quickly 'flip' or position an object in a new direction. To better illustrate, we will again refer to Window 12 of the sample project.



To flip vertically 

1. Select the object you wish to flip vertically. For this example, select the left triangle object of Window 12.

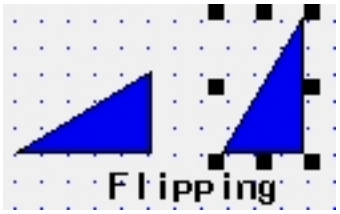
2. From the **Edit** menu, click **Flip Vertical** or click the appropriate icon from the Manager toolbar.



3. From the **Edit** menu, click **Undo** to put the left triangle object back in its original position.

To flip horizontally 

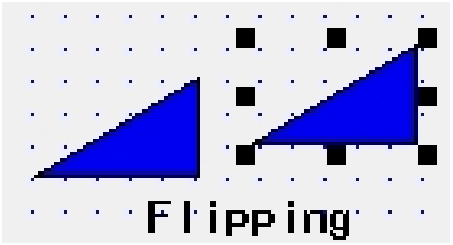
1. Select the object you wish to flip horizontally. For this example, select the right triangle object of Window_12.
2. From the **Edit** menu, click **Flip Horizontal** or click the appropriate icon from the Manager toolbar.



3. From the **Edit** menu, click **Undo** to put the right triangle object back in its original position.

To rotate 

1. Select the object you wish to rotate. For this example, select the right triangle object of Window_12.
2. From the **Edit** menu, click **Rotate 90 degrees** or click the appropriate icon from the Manager toolbar.



To Lock an Object 

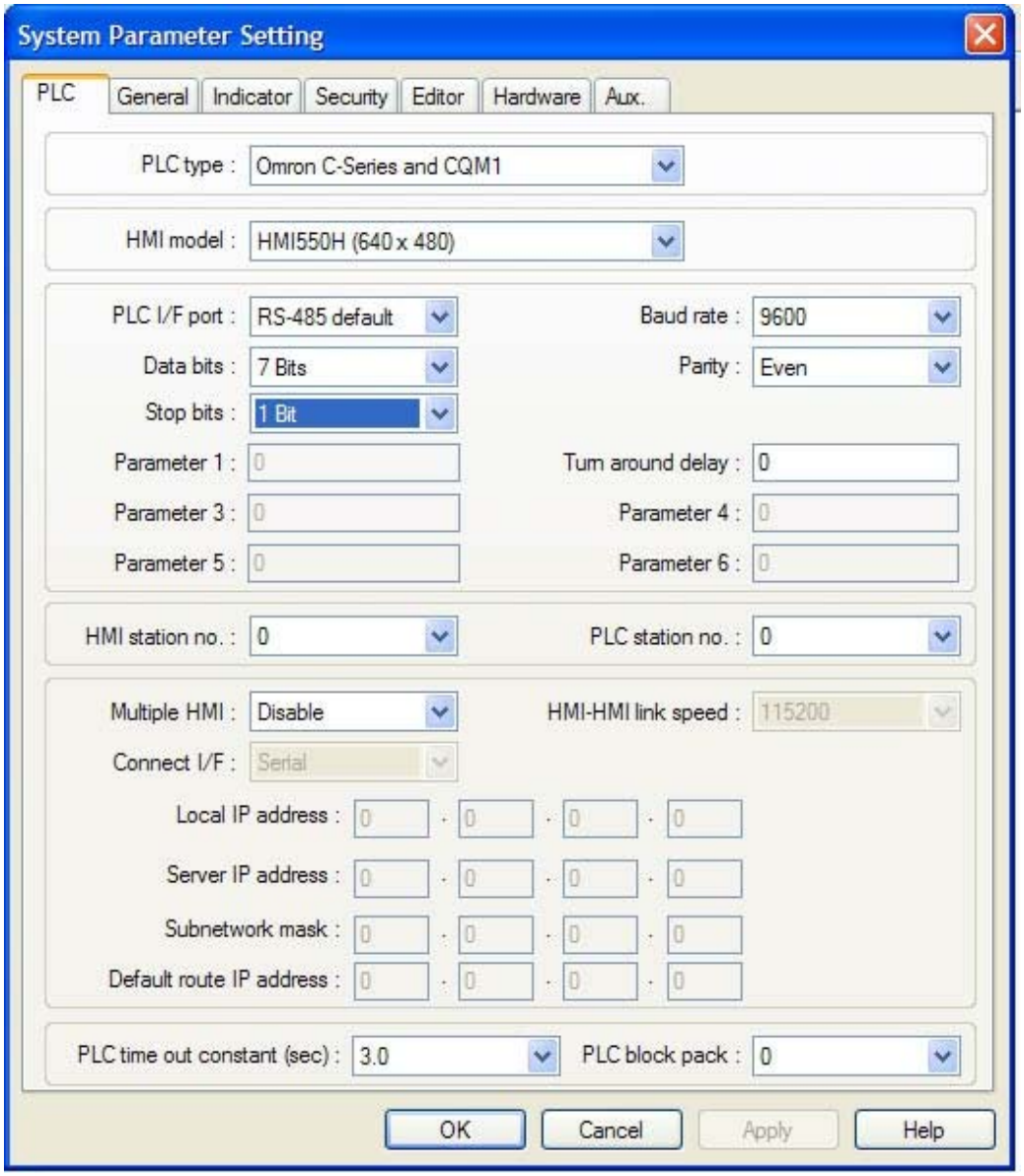
Each object position and size can be locked by using the Pin button. When the object is locked, its position and size cannot be changed.

1. Select the object you wish to lock.
2. From the **Edit** menu, click **Pinned** or click the appropriate icon from the Manager toolbar.
3. To unlock the object, deselect **Pinned** from the **Edit** menu or click the appropriate icon from the Manager toolbar.

General Settings

The final section of this chapter shows all the settings or parameters that can be configured using EasyBuilder. From the **Edit** menu, click **System Parameters**. The System Parameter Setting dialog box appears. The dialog box has six tabs: PLC, General, Indicator, Security, Editor, Hardware, and Aux.

PLC Settings Configuration



To configure PLC settings

1. Select the appropriate *PLC driver* from the **PLC** pull-down menu.

You can "export" a project created for one PLC brand to a different PLC brand by loading the project file into EasyBuilder, then entering a different PLC type in this box. EasyBuilder will go through your entire project and attempt to change each reference to PLC data registers to a logical selection for the new PLC protocol; however, se strongly recommend that you review these changes to assure that they are satisfactory.

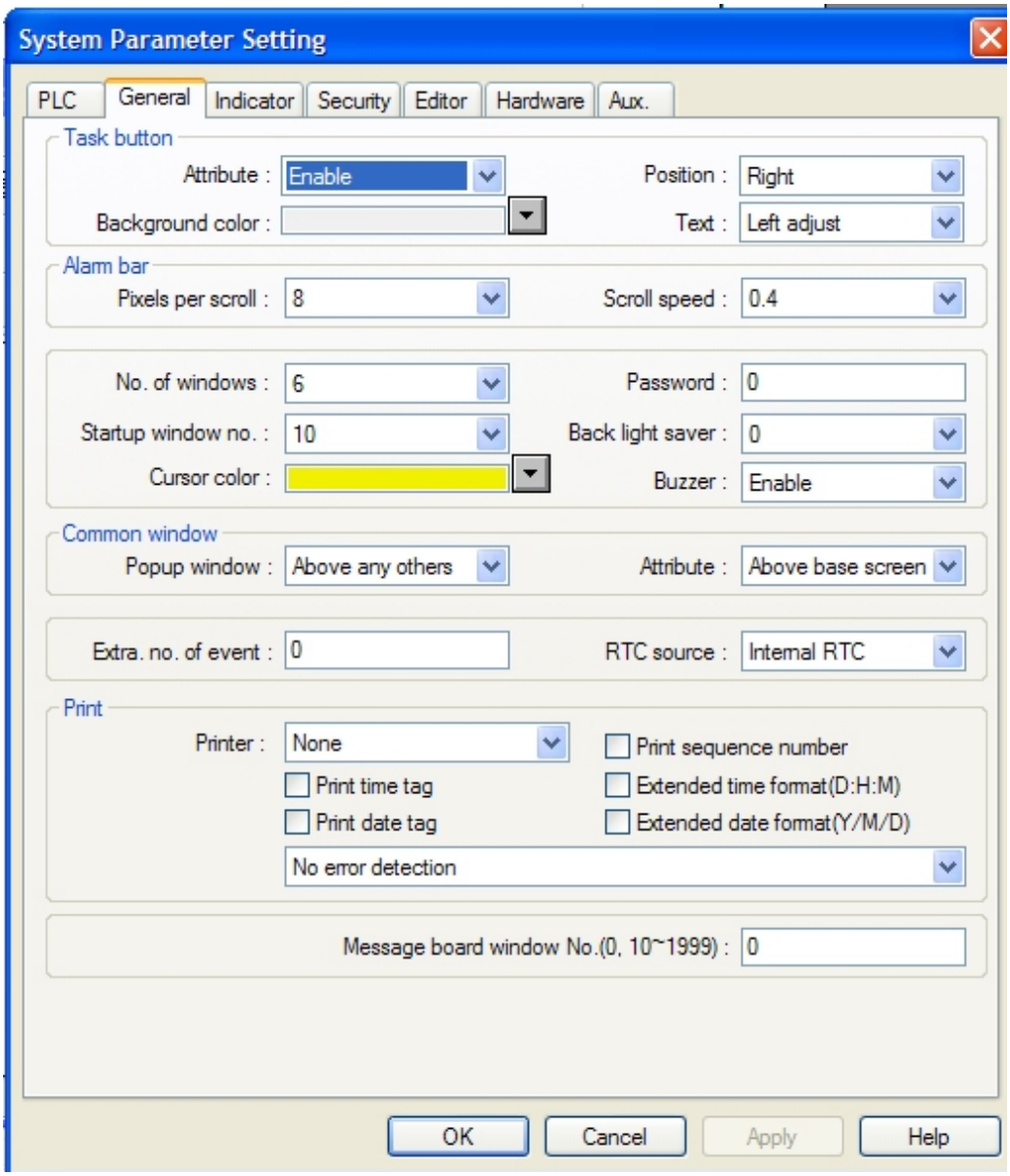
2. If you haven't already done son, select the appropriate *HMI model*: MT506M, MT506C, MT506T, MT506H, MT508C, MT508T, MT510M or MT510H.

You can "export" a project created for one HMI model to a different HMI model. If the original project was created for the color HMI, then changing to the 4-shade blue version causes EasyBuilder to select the closest monochrome shade for each color used. If the original project was created for the 4-shade blue HMI, changing to the color version causes EasyBuilder to select a standard set of colors to represent each blue shade.

3. Set the *serial port type* the HMI will use to communicate with the PLC.
   - **RS232**: Uses the HMI's 9-pin female connector labeled *PLC[RS232]*.
   - The following selections use the HMI's 9-pin male connector labeled *PLC[RS485] PC[RS232]*.

   RS485 default: The selected protocol determines 2 or 4 wire.
   RS485 4W: Forces RS485 4-wire configuration.
   RS485 2W: Forces RS485 2-wire configuration.
   Ethernet: Uses the RJ45 10BaseT connector.
4. Select *baud rate*, *parity*, *data bits*, and *stop bits*. Use Weintek' Controller Information Sheets or the PLC manufacturer's operations manual for information on the communications parameters required by the PLC. These parameters have to match the PLC settings.
5. Parameter 1 and 2 are used to configure a delay time for the HMI. The time set on these parameters is the time the HMI will wait before sending another request for information. Parameter 1 is used in the Protocol level and Parameter 2 is used at the Operating System level. Refer to the Controller Information Sheets if these parameters are required. Parameters 3-6 are not used.
6. Set the *HMI Station No*. used by some protocols that require the HMI to have a unique network address. Consult your Controller Information Sheets for information on the protocols to which this parameter applies.
7. Set the *PLC Station Number* or *Node Address* of the PLC with which the HMI will communicate.
8. **Multiple HMI and HMI-HMI link speed:** Used when connecting multiple HMIs to a single PLC. Disable this setting if you are connecting only a single HMI to a PLC. For more information on connecting multiple HMIs, see *Connecting Multiple HMIs to one PLC* at the end of this chapter.
9. **Connect I/F:** Allows selection of the method by which multiple HMIs can be daisy-chained. The options are *Serial* or *Ethernet*. If *Ethernet* is selected, configure the *Local* and the *Server IP addresses*. For more information, see *Connecting Multiple HMIs Using the Ethernet Port* at the end of this chapter.
10. **PLC Block Pack:** Determines how many words are read from the PLC in one communication cycle. For more information on this feature, see *PLC Block Pack* at the end of this chapter.

General Settings Configuration



1. **Task button attribute:** Used to *Disable* or *Enable* the Task Bar window. The position field determines if the buttons on the window are located on the *Left* or *Right* on the HMI screen. When the Task Bar is enabled, it allows access to the Fast Selection window. The background color and text alignment can also be configured. For more information on the Fast Selection window, consult *Chapter 4 - Creating Windows.*
2. **Alarm Bar Pixels per Scrol**l: Selects the *number of pixels* that the text on the Alarm bar scrolls to the left on each scroll. The speed of the scroll can also be controlled.
3. **No. of Windows:** Selects the maximum number of pop-up windows that can be displayed simultaneously. The default value is 6. The options are 2-6.
4. **Password:** Used to set the *numeric password* that prevents unauthorized uploading of a project. To disable Password protection, set the password to 0. This is the default value. By using this feature, the HMI programmer can prevent another person from using Ezware to upload a project that is stored in the HMI and download it to another HMI.

To Use Password Protection

1. Once the project is downloaded to the HMI, the password becomes active.
2. When you try to upload the project from the HMI to your computer, EasyManager will prompt you to enter a password. If the password is entered correctly, then EasyManager uploads the project file. If the password is incorrect, then an error message appears and the upload process is halted.

- **Startup Window No**.: Used to select which screen will be displayed when the HMI is powered up or reset. The default setting is the initial window that is created with each new project (Window #10). The startup window must be a full-sized window.
- **Backlight saver:** Allows the ability to shut off the HMI's backlight. The number of minutes of inactivity that the HMI will allow before the backlight is shut off can be programmed here. Setting the value to 0 disables the backlight saver. The backlight saver is disabled when the HMI operator touches the screen.

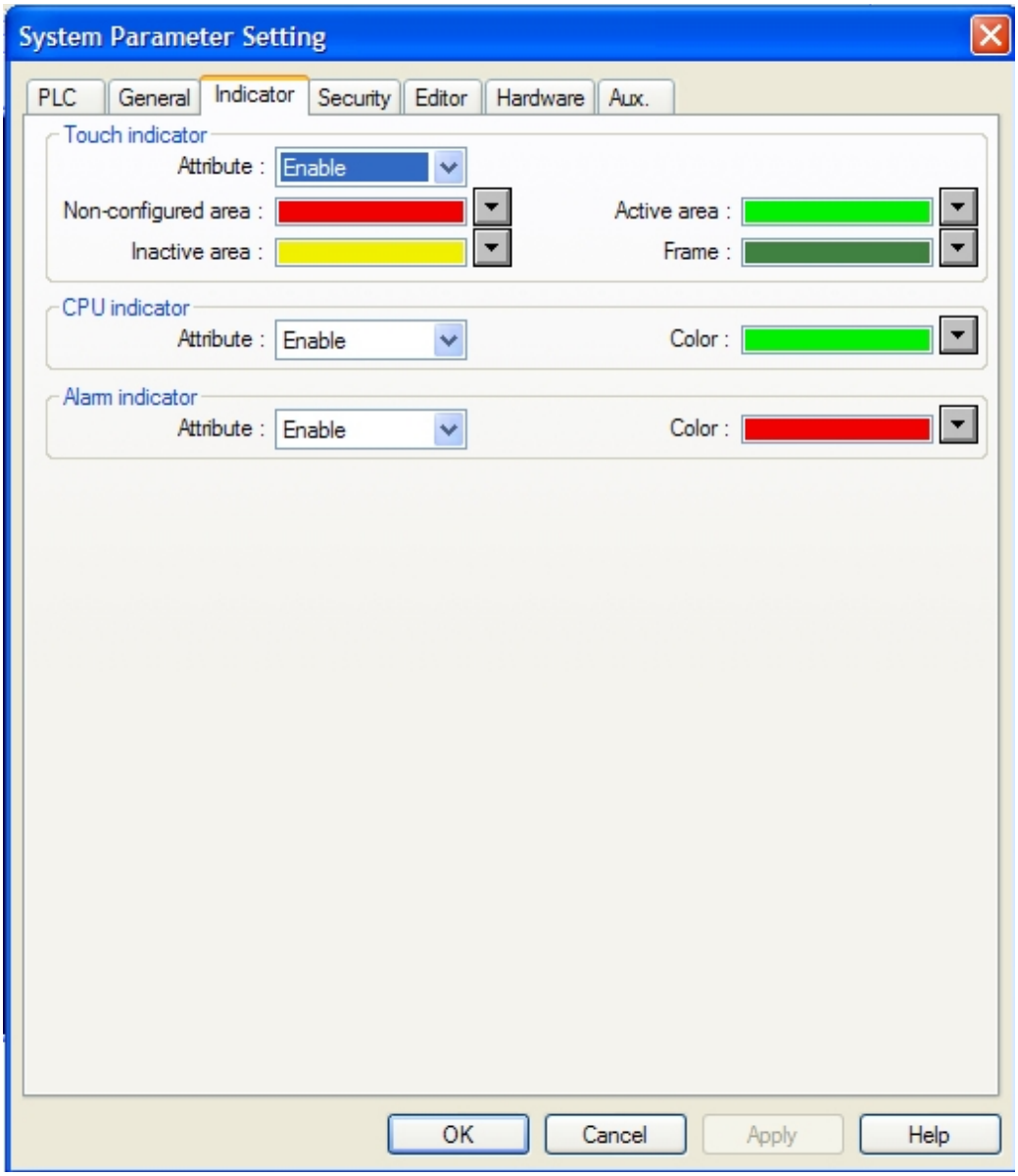
When the backlight is off, any touch of the HMI screen will only activate the backlight, even if the operator presses an active object such as a function key. Once the backlight is on, then the active object will respond to additional key presses.

- The backlight can also be activated by using the **PLC Control Object**. For more information on this feature, see *PLC Control Object.*
- **Cursor color:** Sets the color of the cursor during a data entry operation.
- **Buzzer:** Enables or disables the HMI's buzzer. By default, the HMI sounds a key clock every time the HMI operator touches the screen where an active touch screen object resides. To disable the buzzer, select *disable* on the pull down menu.
- **Pop-up window:** Determines how a pop-up window will behave when placed on a Common window.
  - **Normal:** The pop-up window is displayed when called, and any subsequent windows will be placed in front of it.
  - **Above Any Others:** The pop-up window is displayed when called and remains in front regardless of other subsequent windows displayed.
- **Attribute:** Determines how the common window is drawn.
  - **Below Base Screen:** The base window is placed in front of the common window, so the common window appears behind the base window.
  - **Above Base Screen:** The base window is placed behind the common window, so the common window appears in the front of the base window.
- **Extra No. of Events:** Specifies how many extra events may be placed in the Event Log, beyond the default limit of 200 events. For more information on the Event Log feature, see *Chapter 11 - Capturing Alarms and Events.*
- **RTC Source:** Selects whether the HMI Local Words or its own internal RTC are used for displaying time and date information. The real-time clock is battery-backed with a rechargeable battery. In the absence of power, the battery will retain the clock settings for approximately six months. The battery is automatically recharged whenever the HMI is operating, and has a life of approximately five years.
  - When **Internal RTC** is selected, the time and date information is available to the HMI through certain reserved words in the HMI's recipe area. These addresses can be read from or written to using any of the HMI objects that are word-based, such as Numeric Data, Numeric Entry, Word Lamp and Set Word. Note that Data Type is BCD.
  - When **Local Word** is selected, use the Data Transfer Object to send the PLC's internal RTC to the HMI's Local Words. For more information on the HMI's reserved Local Words, see the section *Using Internal Data Memory of HMI.*
- **Print:** Allows configuration of several settings when using the printing capabilities of the HMI. For more information, see *Chapter 15 - Using a Printer with the MT508/550.*

- **Message Board Window No.:** Allows selection of the number of the base window which will be used as the message board. A setting of 0 disables the message board. Use a number of 10 or higher for this setting. For more information on configuring a message board, see *Chapter 5 - Creating Windows.*

Indicator Settings Configuration

The indicator tab is used to configure different parameters associated with the task bar. The task bar contains three optional indicators; Touch, CPU and Alarm.



The Touch Indicator

- **Attribute:** Enables or disables the touch indicator.
- **Non-configured area:** Sets the background color of the touch indicator when a non-configured area on the touch screen is touched.
- **Inactive area:** Sets the background color of the touch indicator when an inactive area on the touchscreen is touched. An inactive area is an area that contains a touchable object that is currently disabled.

- **Active area:** Sets the background color of the touch indicator when an active area on the touchscreen is touched. An active area is an area that contains a touchable object.
- **Frame:** Sets the color of the circle containing the touch indicator.

The CPU Indicator

- **Attribute:** Enables or disables the CPU indicator.
- **Color:** Sets the color of the CPU indicator.

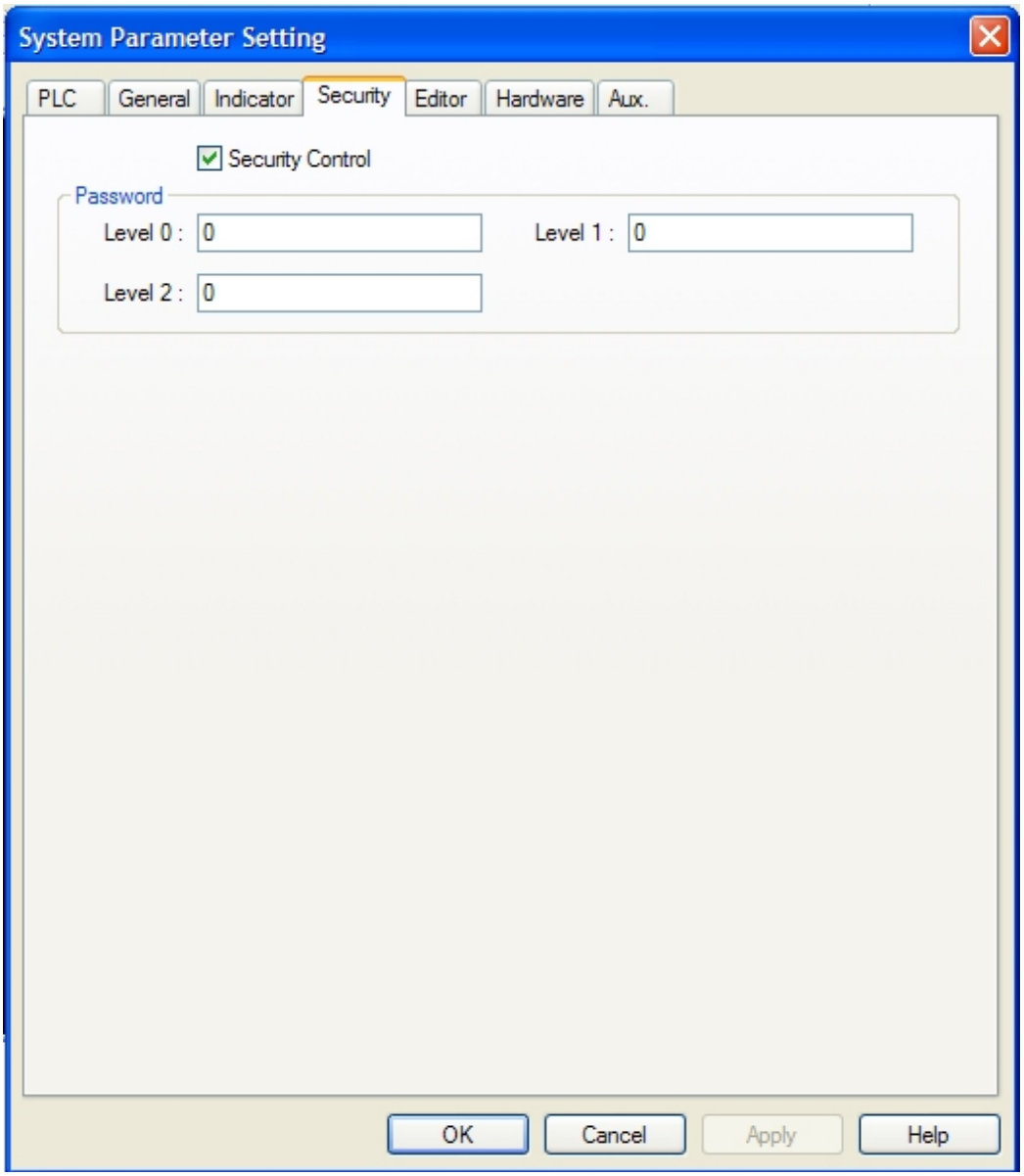
The Alarm Indicator

- **Attribute:** Enables or disables the alarm indicator.
- **Color:** Sets the color of the alarm indicator.

For more information on the task bar, consult Chapter 5 - Creating Windows

Security Settings Configuration

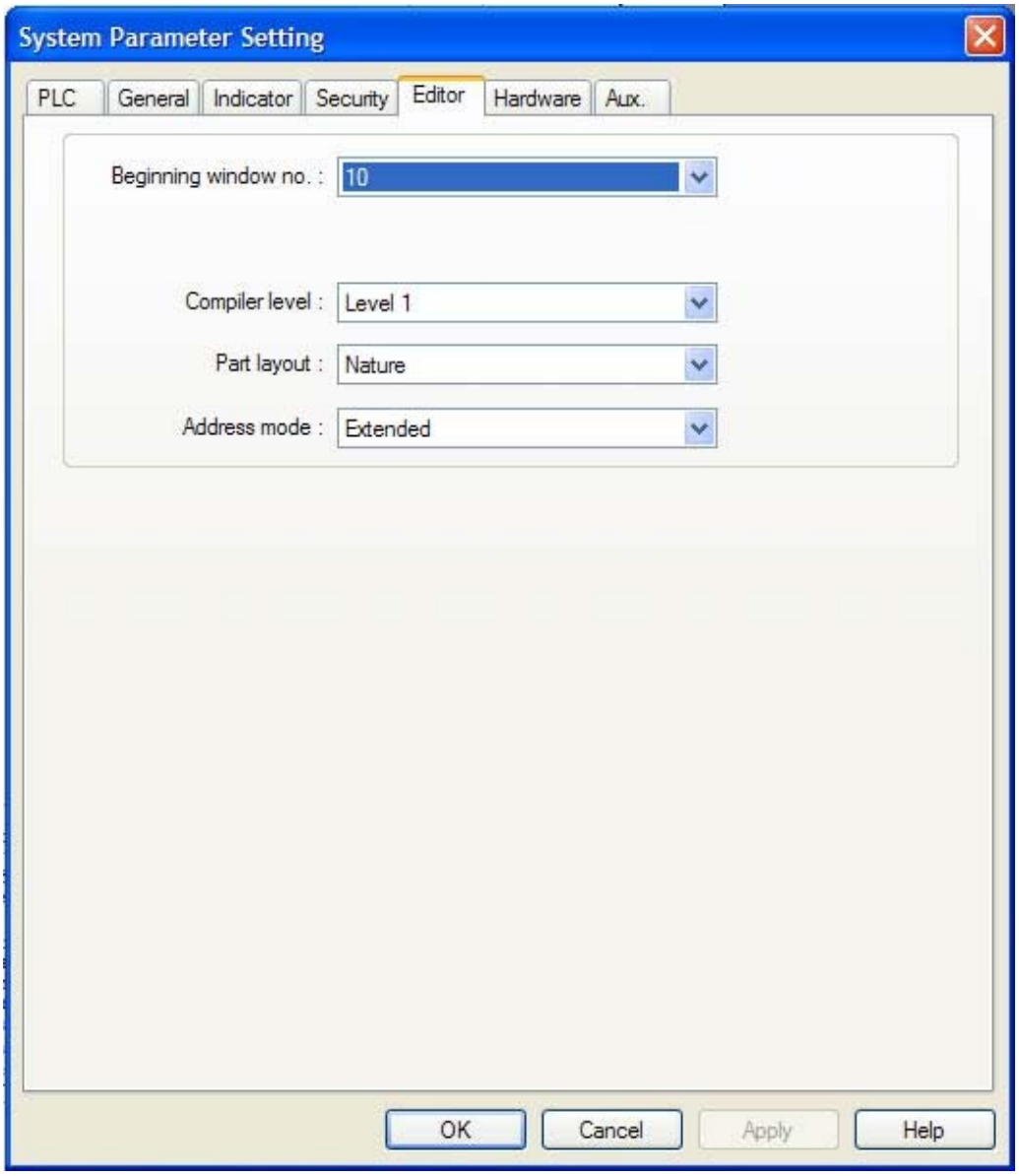
The **Security** tab is used to configure the security features in the HMI. The HMI supports three levels of security. The security features apply only to base windows. Pop-up windows or objects on the screens do not support security.



- **Security Control:** Enables the HMI's security features. The three password fields are not visible if the box is not checked.
- **Password Level 0** is used to enter the numeric password for access to level 0 (the lowest) security. Level 0 is the default security level. This field is not visible if the security control box is unchecked.
- **Password Level 1** is used to enter the numeric password for access to level 1 (the middle) security . This field is not visible if the security control box is unchecked.
- **Password Level 2** is used to enter the numeric password for access to level 2 (the highest) security . This field is not visible if the security control box is unchecked.

For more information on windows security, consult Chapter 5 - Creating Windows

Editor Settings Configuration



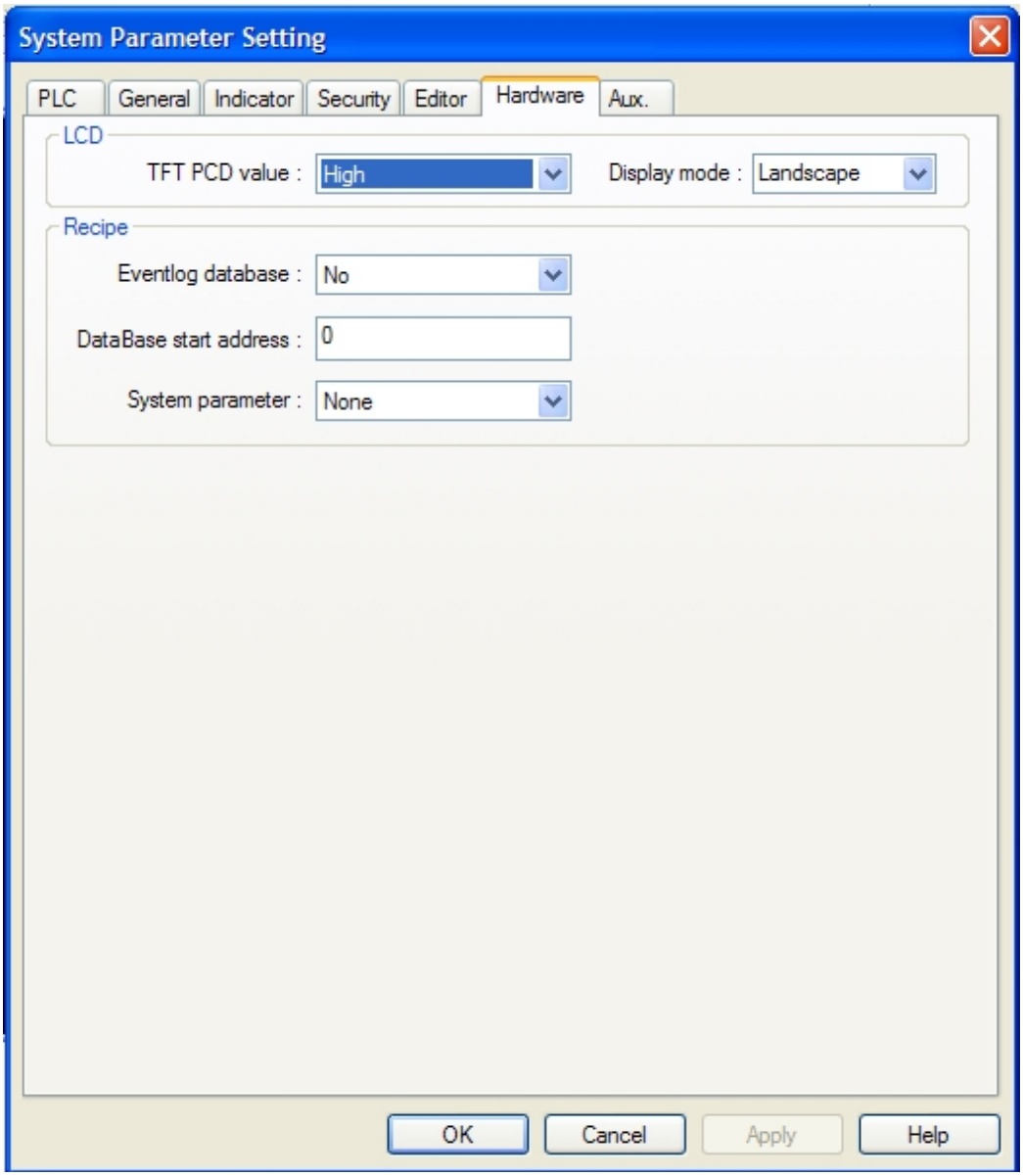
- **Beginning Window No:** Indicates what window-0 number EasyBuilder will start using when numbering windows. When the project is first created in EasyBuilder, the software automatically creates a beginning window. EasyBuilder can be set to start numbering windows that are created, starting with the initial window at either 10 or 1. The default and recommended setting is to start at
- 10.

  Compiler Level: Used to set the compiler warning level. Before a project can be downloaded to the HMI, the project has to be saved and compiled. EasyBuilder can generate two levels of compiler messages. Level 0 is the minimum level and generates only error messages. Level 1 is the highest level, and generates both error and warning messages regarding part and PLC address
- size mismatch. Level 0 is the default setting.

  The **Part Layout** setting allows you to control the level on which objects are displayed during runtime. The **Control** setting causes the HMI to bring objects to the top layer when the object is updated. The **Nature** setting causes the HMI to leave the objects on the layer in which
- they were placed during development.

  Address Mode : Allows the HMI to access more than one PLC. Select *extended* when connecting to multiple PLCs. Select *standard* if connecting to one PLC only. For more information on this feature, see *Networking Multiple Controllers Using Modbus RTU Extended V3 Protocol* at the end

Hardware Settings Configuration

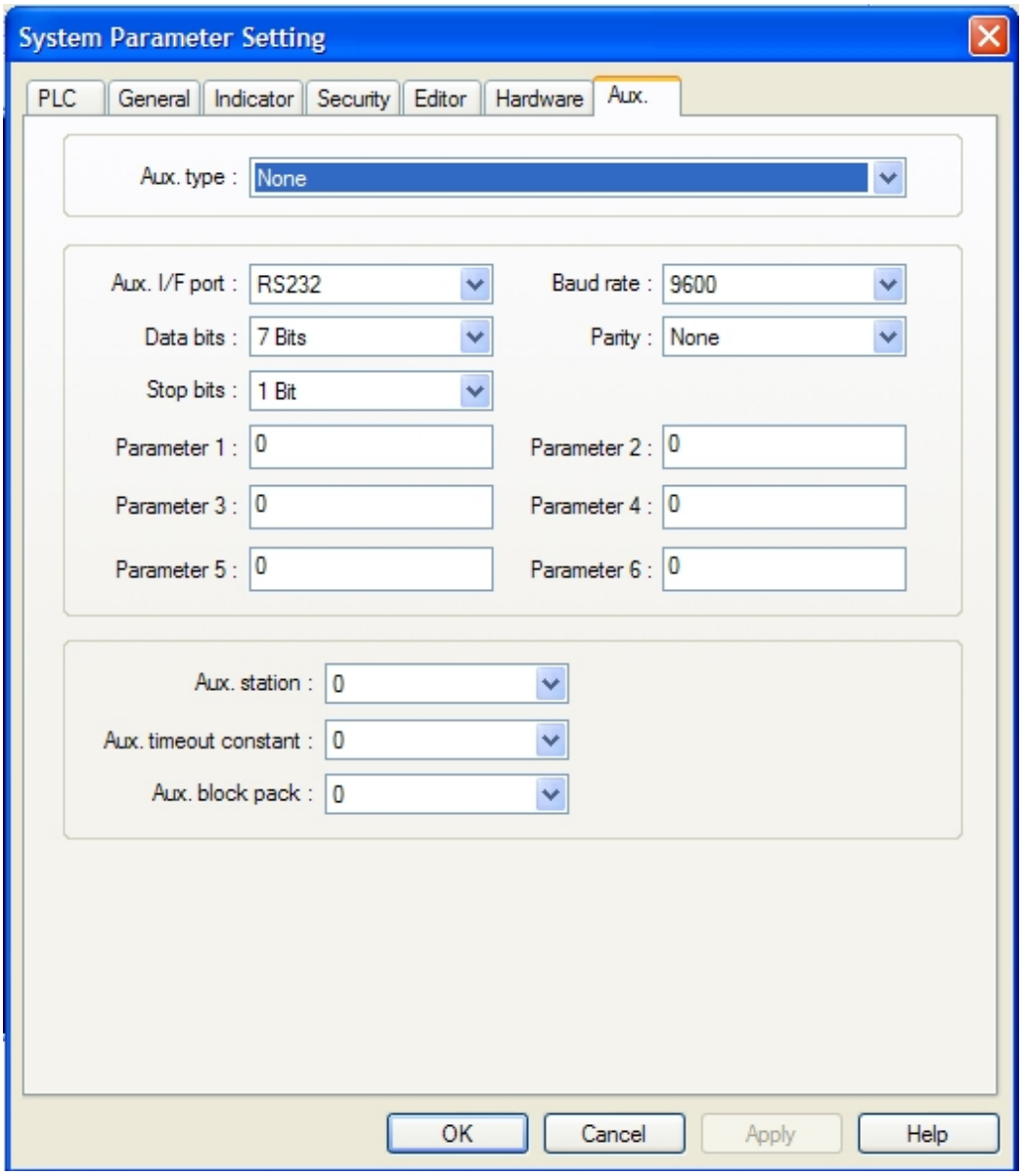


- **TFT PCD value:** Used to adjust the refresh rate of the TFT display. (This option applies only to the HMIs with TFT LCD display type.) This option allows you to select *high* or *low* refresh rate. We recommend using the high refresh setting.
- **Display mode**: Used to select the HMI's display mode. The two options are *landscape* and *portrait.* The default is landscape.
- **Eventlog database:** Enables storing of the event log data in part of the recipe module's non-volatile storage.
- **DataBase start address:** The starting RW address in which Eventlog data is stored. When the memory is full, new Eventlog data is written to this address, overwriting the existing data.
- **System Parameter:** Enables storing of System Parameters in part of the recipe module's non-volatile storage. Data from the General, PLC and Security tabs of the System Parameters dialog can be stored. When this feature is enabled, the HMI reads the data from the non-volatile memory when the HMI is powered up or reset. At runtime, any of these parameters can be

modified with standard objects that write data to the appropriate recipe address. The new values take effect when the HMI is reset.

Aux Settings Configuration

The Aux tab is used to configure the Aux port.



For more information on this feature, see Using the Aux Port at the end of this chapter.

PLC Block Pack

This feature allows the HMI programmer to enhance the update rate of data received from the PLC when the PLC registers displayed on a screen are 'blocked' close together. The default setting is 0. This means that the HMI sends a command to the PLC requesting data for each register that is displayed on screen. For instance, if you have five data fields configured on the HMI to monitor %R1, %R2, %R3, %R4, and %R5 of a GE PLC, then the HMI sends five separate commands to the PLC to update these fields.

This is the slowest method of updating the registers, but it does have the advantage that the update rate does not depend on having the PLC registers all together, (R1-R5). In other words, if monitoring %R1, %R10, %R65, %R156, and %R2048, the update rate would be exactly the same as monitoring %R1-%R5.

The “PLC block pack” feature allows you to take advantage of any groups or ‘blocks’ of registers that allow the HMI to send a smaller number of update commands to the PLC to get the information needed. The HMI will request data for up to 32 contiguous PLC registers depending upon the PLC block pack setting and the addresses of the PLC registers monitored on screen. The PLC block pack # represents the maximum number of ‘skips’ between registers allowed before the HMI considers a PLC register to be part of a different block.

For example, suppose we wish to monitor the following 7 PLC registers on one screen: %R1, %R2, %R4, %R7, %R12, %R14 and %R100.If PLC block pack is set to 0, the HMI sends seven separate commands to get the data. If PLC block pack is set to 1, the HMI checks the spacing between requested registers. If the spacing is more than 1, then a separate command is sent. In this example, the HMI sends four commands- one command to read R1, R2, and R4; one command to read %R7, one command to read %R12 and %R14, and one command to read %R100.

With most PLC protocol drivers, more time is spent with ‘overhead’ data than getting the actual data that you need. With the example above, setting the block pack number to 1 probably reduces the time required to update the PLC data by as much as 40%! So properly setting the PLC block pack can have a major impact on the update rate of the HMI.

If the PLC block pack setting were adjusted to 4 or greater, the HMI sends only two commands- one command to read %R1, %R2, %R4, %R7, and %R12; and another command to get %R100. The update rate is now probably three times as fast!

You might think that the best setting is 10 so why not have the PLC block pack always on 10? However, in some situations, a setting of 10 may actually slow down the update rate. For example, if the PLC registers displayed on the screen are actually %R1, %R10, and %R40 with the PLC block pack is set to 10, the HMI will send two commands: one to get %R1 and %R10, another to get %R40. If the PLC block pack were set to 0, then the HMI would have sent three commands. In this case, however, the three commands probably take less time than the two commands because the HMI has to receive data for ten registers (%R1-%R10) in one command takes more time than the overhead associated with two commands. Therefore, you should experiment with this setting if you find that the update rate is longer than you would like.

Now that you know your way around EZware-500 and its various applications, it is time to start creating your own project. The next seven chapters show you in detail, all of the features that you can use to create a user-friendly touch screen graphics operator interface terminal. Along the way, we will attempt to provide you with clear definitions of each feature and, if necessary, some additional examples of how each feature might be used.

Restart the HMI Automatically after a Project is Downloaded

EasyBuilder can automatically start the application that was just downloaded.

To enable automatic restarting of the HMI

1. From the **Option** menu, select **Window Property**. The **Window Property** dialog box is displayed.
2. Check the **Jump To Application When Download Done** checkbox..
3. Click **OK** to exit the **Window Property** dialog box.

Automatically Save and Compile the Project

EasyBuilder can automatically save and compile the project when downloading or simulating.

To enable automatic save and compile

1. From the **Option** menu, select **Window Property**. The **Window Property** dialog box is displayed.
2. Check the **Automatic Save And Compile At Download And Simulate** checkbox.
3. Click **OK** to exit the **Window Property** dialog box.

Compact Flash

Compact Flash (CF) cards enable one to download an EZware project from a PC to a compact flash card, and then put that card into the HMI, and download the project to the HMI. This eliminates the need for a laptop or PC at the installed HMI's location, and makes downloading the same project to multiple HMIs simple.

Any CF Type1 flash card may be used that has been formatted for the FAT format, and has a memory size equal to or greater than 64MB.

To store the file from the PC to the Compact Flash Card:

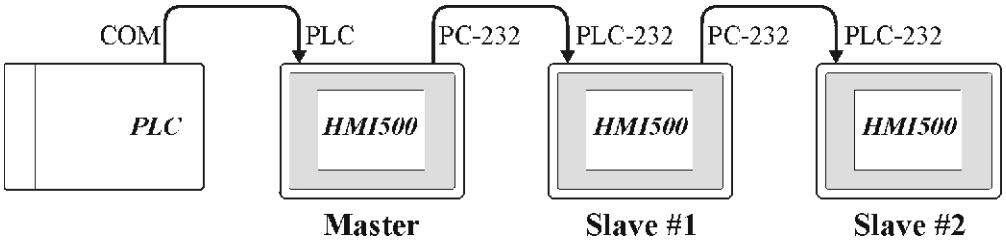
1. Open ImageCF.exe (located in the EZware folder). The following screen will appear.
2. Press the magnifying glass to browse for your project's .eob file. Press open, and ImageCF.exe automatically converts the *.eob file to an "ImageCF.bin" file which will create itself in your EZware directory.
3. Copy the ImageCF.bin file to the Compact Flash card. (Using Windows Explorer, drag the ImageCF.bin onto your Compact Flash Reader device.)
4. Insert the Compact Flash card into an HMI.
5. Find the Dip Switches on the back of the HMI. Set Dip Switch 2 "On," then push the Reset button (or cycle power). The HMI will show some text and at the bottom right side of the screen a green button will appear. Press the green button and the HMI will load the project from the Compact Flash Card and automatically reboot to application mode.

Dip Switch 2 on the back of the HMI should be moved back to the "Off" position after the project has downloaded.

Connecting Multiple HMIs to one PLC

Most of the PLCs on the market communicate to HMIs using a master/slave format. The PLC typically behaves as the slave responding to requests from the HMI for data and not initiating any communications. Therefore, the HMI must continuously poll the PLC for data needed. Some of the PLC protocols do not allow multiple master devices (HMIs) to connect to the PLC. If you want to connect two or more HMIs to a PLC, you often need to purchase multiple communications modules for the PLC, which can be very expensive.

To address this need, the MT5xx has a built-in feature which allows multiple HMIs to connect together in a daisy-chain fashion. One of the HMIs is designated as the 'master' while the other HMIs are defined as 'slaves'. The master HMI is directly connected to the PLC communications port. Each slave HMI then communicates through the master HMI to get data from the PLC.

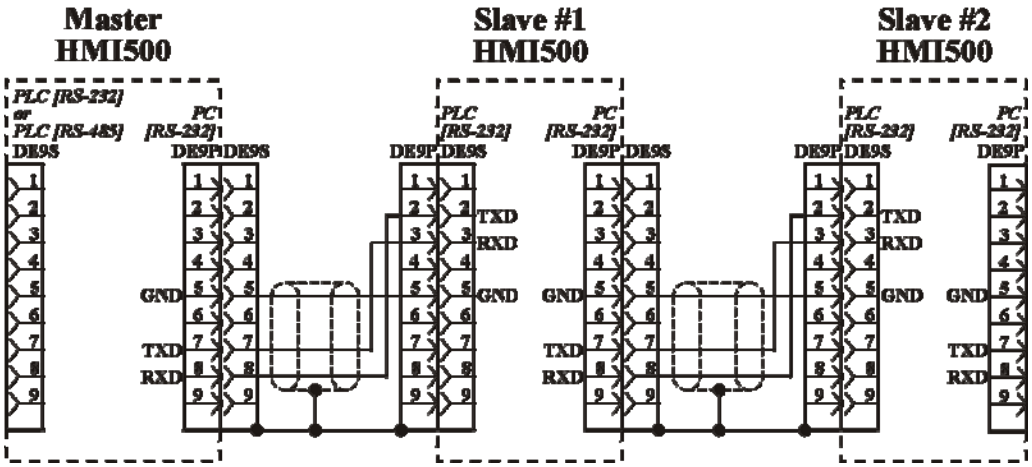


No additional communications modules are required. This feature provides an effective and inexpensive way to connect multiple HMIs to a single PLC.

Hardware Connection

stated above, the HMIs are connected in a daisy-chain fashion using RS232 communications. This means that you can have a maximum of 50 feet distance between HMIs. Also, because connection uses a daisy-chain method,

if one of the HMIs loses power all other HMIs further down the chain will be unable to communicate to the PLC. The HMIs are wired together in the following fashion:



Software Connection

You must configure the HMI that connects to the PLC as the master. All other HMIs should be configured as slave HMIs.

To configure the Master HMI

1. Click the **Edit** menu on the main screen of EasyBuilder.
2. At the bottom of the menu, click **System Parameters**. The Set System Parameters dialog box appears.
3. The dialog box has three tabs: PLC, General, and Indicator. Select the **PLC** tab.
4. Select **Master** from the **Multiple HMI** box.
5. Select either **38400** or **115200** for the **HMI-HMI link speed**. Make sure this setting matches the settings for the slave HMIs.
6. Select a unique station number in the **HMI station no.** box.
7. Click **OK** to save the System Parameters settings and return to the main screen of EasyBuilder.

To configure the Slave HMI

1. Click the **Edit** menu on the main screen of EasyBuilder.
2. At the bottom of the menu, click **System Parameters**. The Set System Parameters dialog box appears.
3. The dialog box has three tabs: PLC, General, and Indicator. Select the **PLC** tab.
4. Select **Slave** from the **Multiple HMI** box.
5. Select either **38400** or **115200** for the **HMI-HMI link speed**. Make sure this setting matches the setting for the master HMI.
6. Select a unique station number in the **HMI station no.** box for each slave HMI.
7. Click **OK** to save the System Parameters settings and return to the main screen of EasyBuilder.

Sharing Data Between the Master HMI and Slave HMI

Each HMI can be programmed to access data from the PLC as though it were the only HMI connected to the PLC. The master HMI will poll each slave HMI connected and allows each HMI to retrieve data from the PLC. You can also exchange data between the slave HMIs and the master HMI. Any local word registers (LW) or local bits (LB) located in the master HMI can be accessed and written to by the slave HMIs. Do this by using the Ms_LW and Ms_LB memory registers. For example, if the master HMI is storing data in LW10, the slave HMI can access that data by using Ms_LW10.

If the slave HMIs and the master HMI are both accessing the same areas of memory in the PLC, it becomes more efficient to program the master HMI to access the PLC memory than create a Data Transfer Object to copy the PLC data to LW and LB registers inside the master HMI. The slave HMIs can then access the PLC data by using the Ms_LW and Ms_LB registers. The biggest bottleneck in communications is trying to get all of the needed data from the PLC. By doing this, you greatly reduce the amount of time required to get the information from the PLC.

Connecting Multiple HMIs Using the Ethernet Port

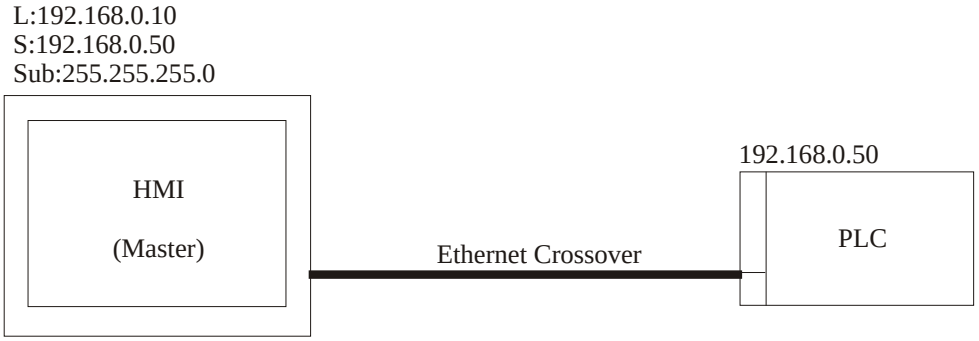
The MT5xx HMIs are equipped with Ethernet ports. One, or several HMIs can be connected to a PLC that uses the Modbus TCP/IP protocol. A single HMI may be directly connected to the PLC Ethernet port via a crossover Ethernet cable, or multiple Master HMIs may be connected to a PLC using an Ethernet switch and standard Ethernet cabling. Each slave HMI can communicate through the master HMI to get data from Master HMI local memory and the PLC. No additional communications modules (other than possible network infrastructure such as routers, hubs and switches) are required, providing an effective an inexpensive way to connect multiple HMIs to a single PLC.

The PLC must use a Modbus TCP/IP protocol driver if it is to be connected to the HMI Ethernet port. PLCs that use other protocols such as Industrial Ethernet or AB Ethernet IP are not supported by the MT5xx touchscreens. Only Modbus TCP/IP is supported.

Hardware Connections

There are several wiring scenarios for connecting HMI touchscreens to a PLC. Most depend upon individual needs.

The simplest connection (with example network IP addresses) is a point-to-point connection, such as the one illustrated below:

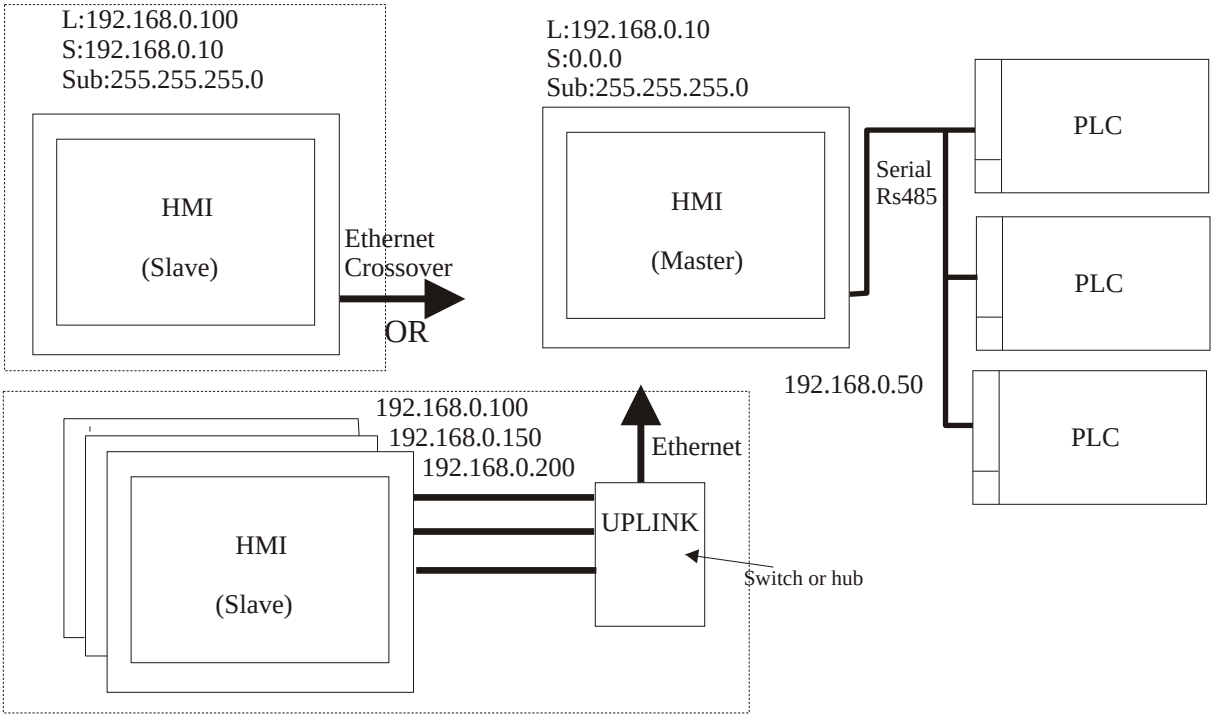


This allows one HMI touchscreen to be the master and the PLC to act as the slave device. Only one PLC can be addressed in this type of configuration.

Multiple PLCs

If more than one PLC is to be addressed by a single HMI, use the HMI RS485 port and use the Modbus RTU Extend V3 protocol. One slave HMI may be connected to the master via a crossover Ethernet cable or several Slave

HMIs may be connected to the master using an Ethernet switch or hub device. If a hub is used, use standard Ethernet cabling and connect the HMI master cable into the UPLINK port.

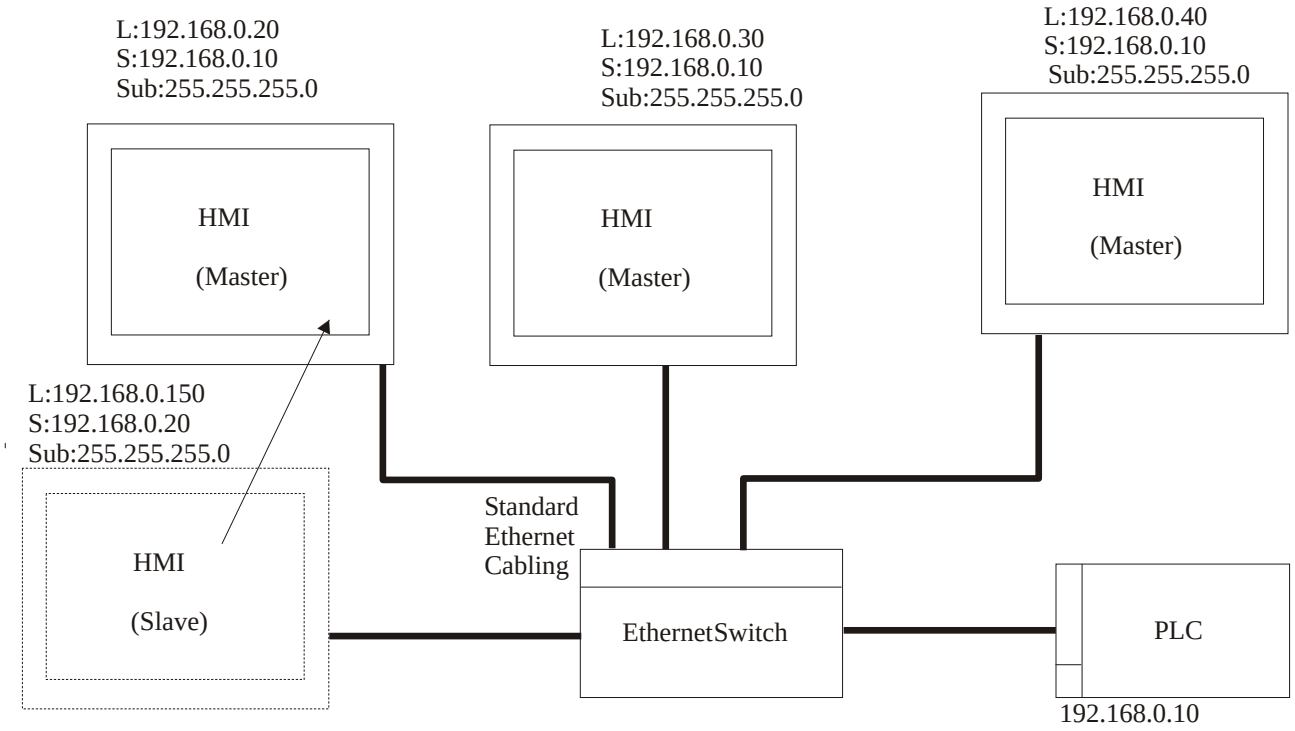


To configure the scenario above , select "Modbus RTU Extend V3" for the PLC type. PLC I/F should be RS485 default. Select Multiple HMI (Master/Slave) and connect Ethernet. Under the {**Editor**} tab of the **system parameters**, select Address Mode: **Extended**.

Multiple HMI Masters

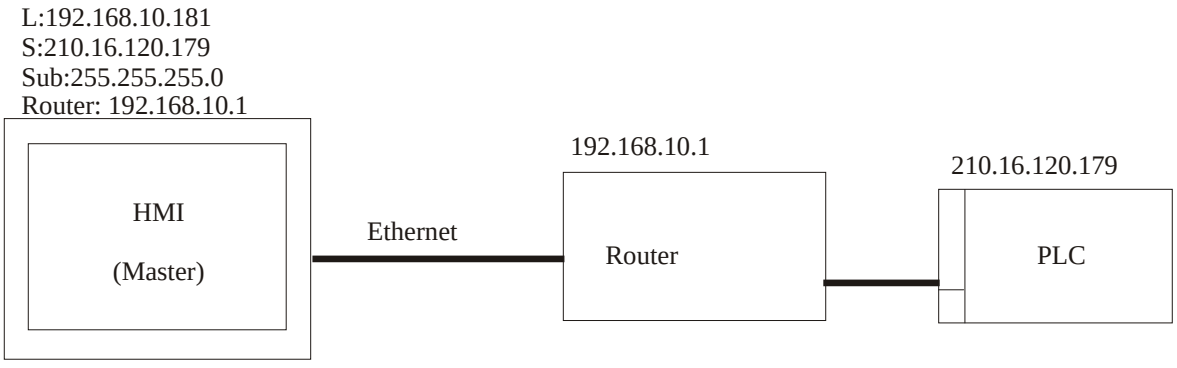
If several HMIs need to be connected to one PLC and the HMIs need to be independent of one another (each can be

disconnected without interrupting the other), then they can be connected as individual master devices using an Ethernet switch, as illustrated below:



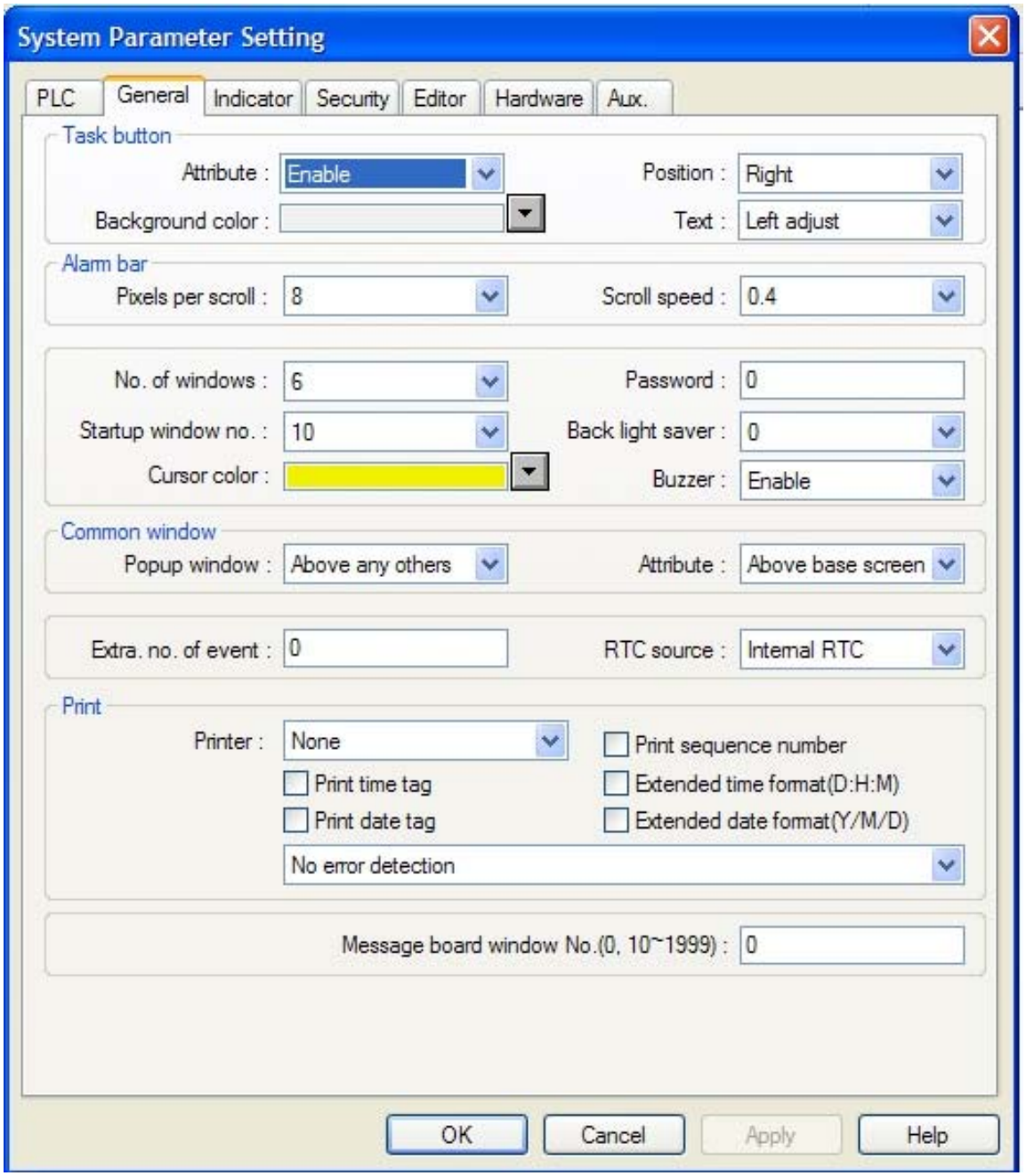
This configuration can only support one PLC, and all master HMIs must use the same protocol and the same server address for the PLC. A master HMi will not be able to see local memory or status of any other master HMI. A slave HMI can be connected to monitor/control a specific HMI master. A slave can control and view the internal memory and status of a master HMI.

If a router is to be used for connecting to a different network, then enter the router IP address and configure the router to address the PLC on its network. See the illustration below:



Software Configuration

In the **Edit-System Parameters** -{**PLC**} tab, configure as follows:



You must configure the HMI that connects to the PLC as the master. All other HMIs should be configured as slave HMIs, unless an Ethernet switch is used, in which case the HMIs may all be masters with separate local IP addresses, and all have the same Server (PLC) address.

To configure Master HMIs to talk with a PLC over the Ethernet port:

1. Click on **Edit-System Parameters** and select the {**PLC**} tab.
2. Select PLC Type as **Modbus RTU TCP/IP**.
3. Select PLC I/F port as **Ethernet.**

4. For the multiple HMI box, select **Disable** if there are no slave HMIs. Select **Master** if any HMI is to be a slave unit, and, if so, select **Ethernet** for Connect I/F.
5. Select a *unique IP Address* in the Local IP Address: boxes.
6. Enter in the *IP Address for the PLC* in the Server IP Address: boxes. Note: Whatever the network addresses are, the numbers (octets) in the first three boxes must match for the Local IP and the Server IP.
7. Enter **255.255.255.0** for the Subnetwork Mask.
8. If there is a router that will be between the HMI and the PLC, enter the *router IP address* in the Default Route IP address section.

To configure the Slave HMI if there is an Ethernet connected Slave HMI Unit:

1. Click **Edit-System parameters** and select the **PLC** tab.
2. Select PLC Type as **Modbus RTU TCP/IP.**
3. Select PLC I/F port as **Ethernet.**
4. For the Multiple HMI box, select **Enable.**
5. Select **Slave** from the Multiple HMI box.
6. Select **Ethernet** for the Connect I/F.
7. Select a *unique station number* in the HMI station no. box for each slave HMI, and click **OK**.

Sharing Data between the Master HMI, Slave HMI and PLC

Master HMI to a PLC

The master HMI will be programmed just as if it were the only HMI connected to the PLC.

Slave HMI to a Master HMI

Any local word registers (LW) or local bits (LB) located in the master HMI can be accessed (read) and written to (write) by the slave HMIs. Do this by using the "Ms_LW" and "Ms_LB" memory registers. For example, if the master HMI is storing data in LW10, the slave HMI can access that data by using a device type: **Ms_LW** and device address: **10**. Multiple HMI is set to **Master**. Connect I/F is set to **Ethernet**.

Slave HMI to the PLC

The slave HMIs may access data in the PLC in two ways:

1. It may write to the local words/bits in the master HMI (via Ms_LW and Ms_LB registers), and those values may be transferred to the PLC by setting up a data transfer object in the master HMI program. Note: If the master and the slave HMIs are accessing the same areas of memory in the PLC, this method is more efficient because the time required to get information from the PLC is reduced.
2. It may read/write directly to the PLC addresses (as though it were the only HMI connected to the PLC). Note: This method may result in inconsistent Master/Slave communications. Weintek recommends using method #1.

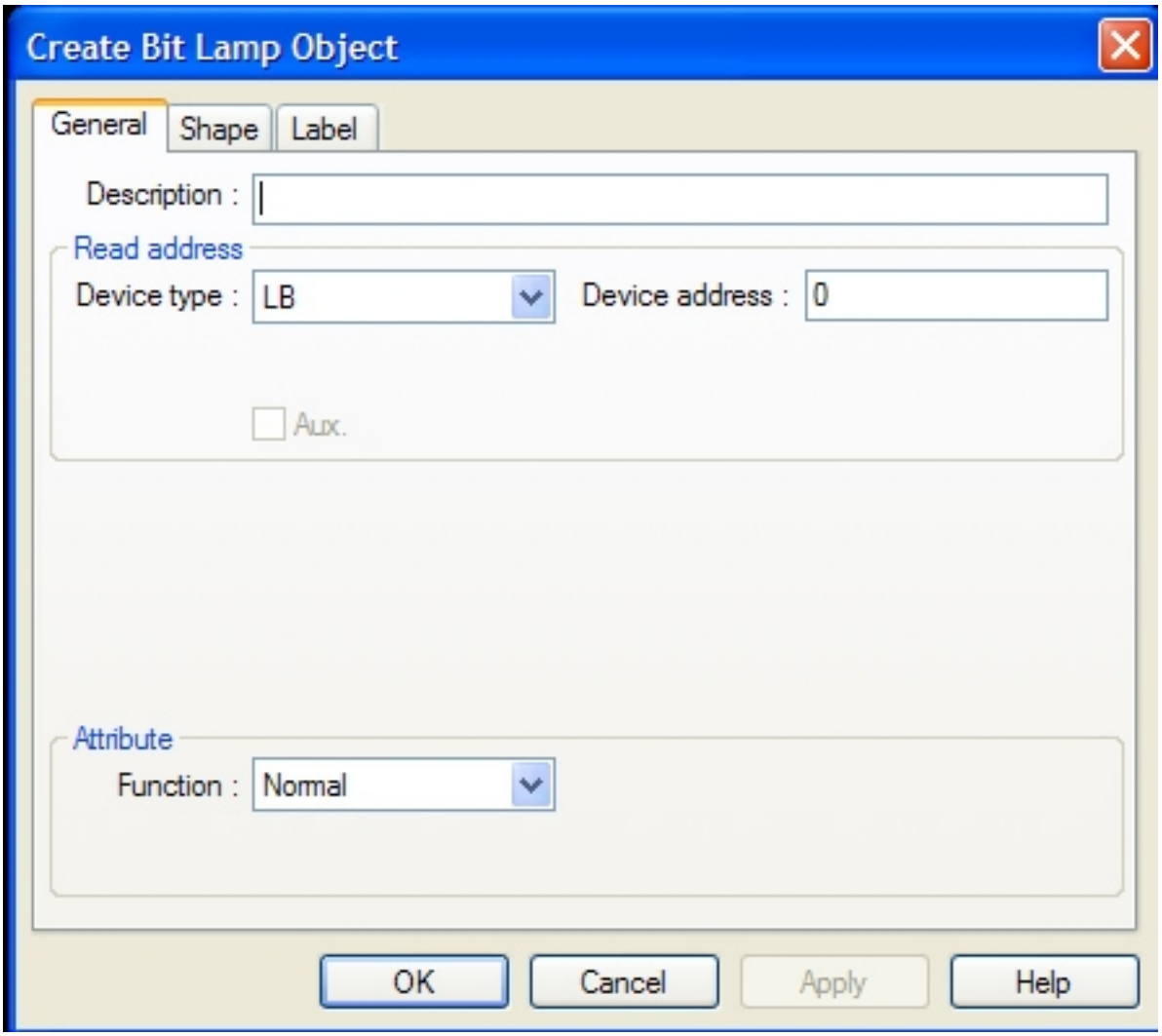
Networking Multiple Controllers

The MT5xx provides capability of accessing data in multiple controllers from a single master station. The MT5xx can address each object to its respective controller node, giving operators the ability to seamlessly interact with an entire multi-drop network without changing either screen or station numbers. This allows the information to be organized in the most logical way rather than being clumped by controller address, which may not be anywhere near ideal for the operator. In addition, directly accessing the necessary data from each controller eliminates the need to use a data concentrator PLC to present a single point of access. With the inherent noise immunity and line-length capability of RS485, you can view information and control equipment from anywhere in the plant at one convenient location.

Operators can access up to 16 slave controllers by setting the **Address Mode** in the **System Parameter-Setting-Editor** dialog box to **Extended**. Each object on the HMI is given a slave address, delineated by a # sign, followed by the register address.

To configure a Bit Lamp object using Extended Addressing:

1. From the **Parts** menu, click **Bit Lamp**. The Create Bit Lamp Dialog appears.



2. In the **Read address** frame, select the *PLC coil*. For **Device address**, enter the *PLC node address* followed by a **#** sign and the *register address*. For this example, the PLC node=1 and the register address=100.

The system may be wired with either a 2-wire or 4-wire cabling scheme, with a single signal return wire in either case. This is selected in the **System Parameter Setting** dialog on the **General** tab.

In genera, a 4-wire system is preferred, since it prevents the possibility of the Master (HMI) and the Slave (controller) units interfering with each others' transmissions. There is a user-adjustable polling delay.

To set the user-adjustable polling delay:

1. Select **Edit-System Parameters** from the main menu. The System Parameter Setting dialog appears.
2. Select the **PLC** tab.
3. For **Parameter 1**, enter a value between *1-999* (mSec).
4. It is best to start with a large delay until communication is established with all of the slave controllers, then decrease it until a minimum reliable setting is reached. This is especially

true of 2-wire configurations, as the HMI may reissue a command if the reply is not processed quickly enough.

5. For **PLC I/F port**, select *RS485 4W or RS485 2W*.

Managing Communication Failures

When communicating with multiple devices over a network, it is possible that one or more of the devices may fail without affecting other devices on the network. When this happens, plant personnel may need to be notified of the failure, while still maintaining communications with the rest of the network.

Starting in EasyBuilder v 2.60, communications failures to any node can be managed by the HMI. A series of local bits monitor and control communications failures on both the main and auxiliary communications ports.

Bits	PLC Node
LB9100-9227	Main port nodes 0-127
LB9228-9355	Aux port nodes 0-127

When a communications failure is detected on a particular PLC node address, the corresponding local bit will be turned off, and communication to that node address is suspended. To restart communications to that node, the bit must be turned back on.

Using the Aux Port

HMI generation 6 (marked-006) models are now equipped with a secondary Aux port for both RS-232 and RS-485 communications. This aux port can be used to operate a secondary PLC. To use the aux port correctly, the user must upgrade their EasyBuilder software to version 2.6.0 or higher. The setup screen for the aux port can be found as a tab option under the **Edit-System Parameters** menu selection.

PLC [RS-232] and PLC [RS-485] + AUX [RS-485] Port

Pin Designations

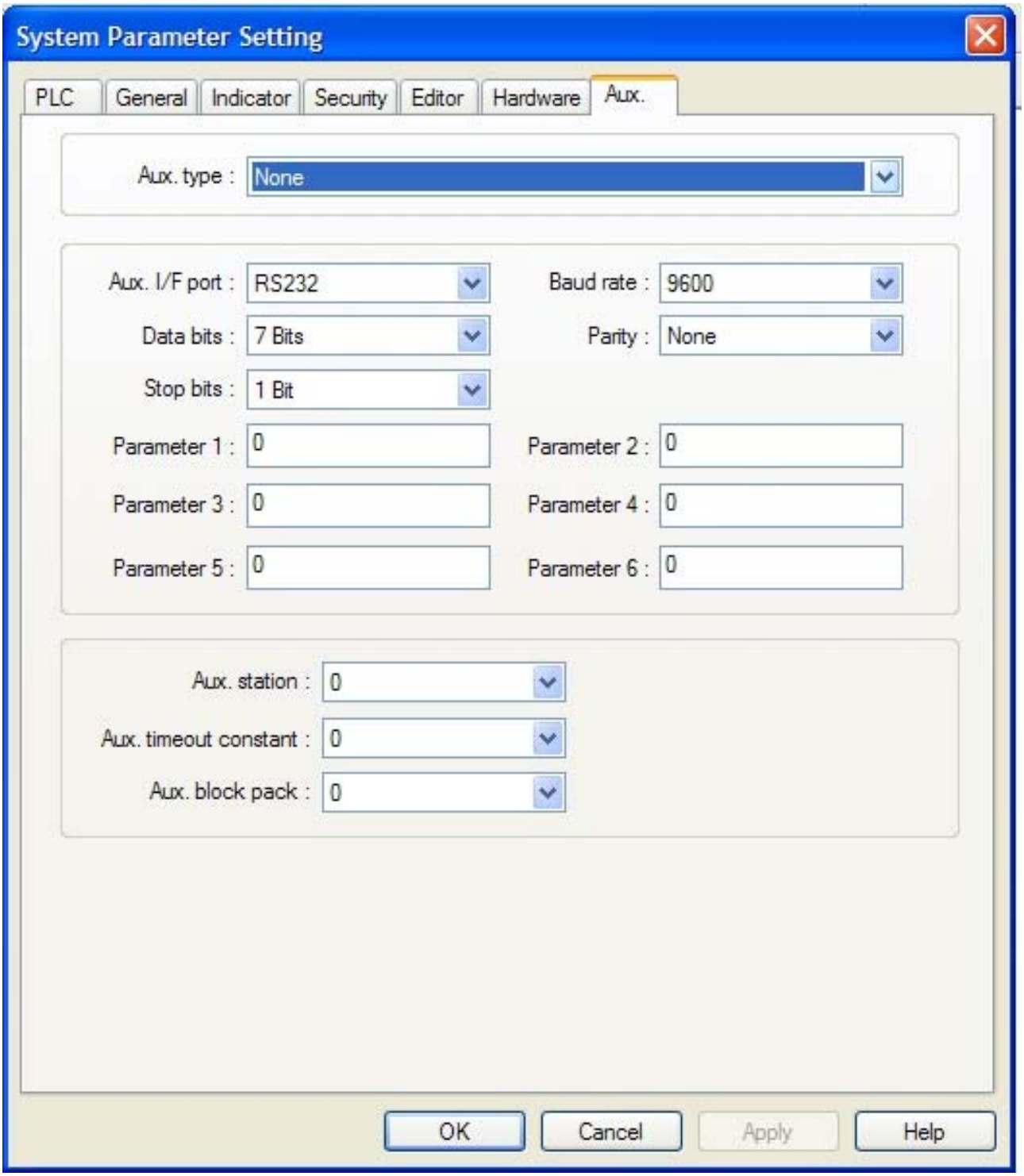
Pin assignment of the 9-pin, female Sub-D, PLC [RS-232] Port

Pin #	Symbol	PLC [RS-232]	AUX [RS-232] only -006 Series
1	Aux_TxD		Transmitted Data
2	TxD	Transmitted Data	
3	RxD	Received Data	
4	Not used		
5	GND	Signal Ground	
6	Aux_RxD		Received Data
7	CTS	Clear to send input	
8	RTS	Ready to send output	
9	Not used		

Pin assignment of the 9-pin, male Sub-D PC [RS-232] and PLC [RS-485] Port

Pin #	Symbol	PLC [RS-485] 4-wire main port	PLC [RS-485] 2-wire main port	AUX [RS-485] (only -006 series)	PC [RS-232]
1	Rx-	Receiver (-)	Xcver_Data (-)		
2	Rx+	Receiver (+)	Xcver_Data (+)		
3	Tx-	Transmitter (-)			
4	Tx+	Transmitter (+)			
5	GND	Signal Ground			
6	Aux_Data-			Xcver_Data (-)	
7	PC_TxD				Transmitted Data
8	PC_RxD				Received Data
9	Aux_Data+			Xcver_Data (+)	

In version 2.6.2, set in **Edit-System Parameters**-{**Aux**} tab dialog set the *Aux port driver* (**Aux type)** and *communications parameter*.



To set up the Aux I/F port parameters

1. Select the *Aux Port driver*. Currently, it only supports the Animatics SMC v. 1.00, Baldor MINT v. 1.03, and the Modbus RTU Extend [AUX v. 3.00] protocols.
2. Select the *hardware port* to use for auxiliary communications. (RS232 port - full duplex, RS-485 - half duplex.
3. Select the *Aux port communications timing parameters.* Set the communications parameters to match your PLC.
4. Set *Data bits*, *Stop bits*, *Baud rate* and *Parity*. These must match your PLC settings.
5. Set parameters 1-6, which are dependent on the individual driver selected.
   a. For Modbus RTU Extend V3 [AUX] protocol:
   - **Parameter 1:** Not used.
   - **Parameter 2: Turn Around Delay (n*mS):** This setting is used to add some delay for RS485 (half-duplex) communications. The number entered is the extra wait time (in 10 mS increments) that the HMI adds before it transmits a retry on the bus. (Range: 0-1000). If RS232 is selected as the I/F port, then this parameter is ignored.
   - **Parameter 3-6:** Not used.

To set Aux ID and general parameters

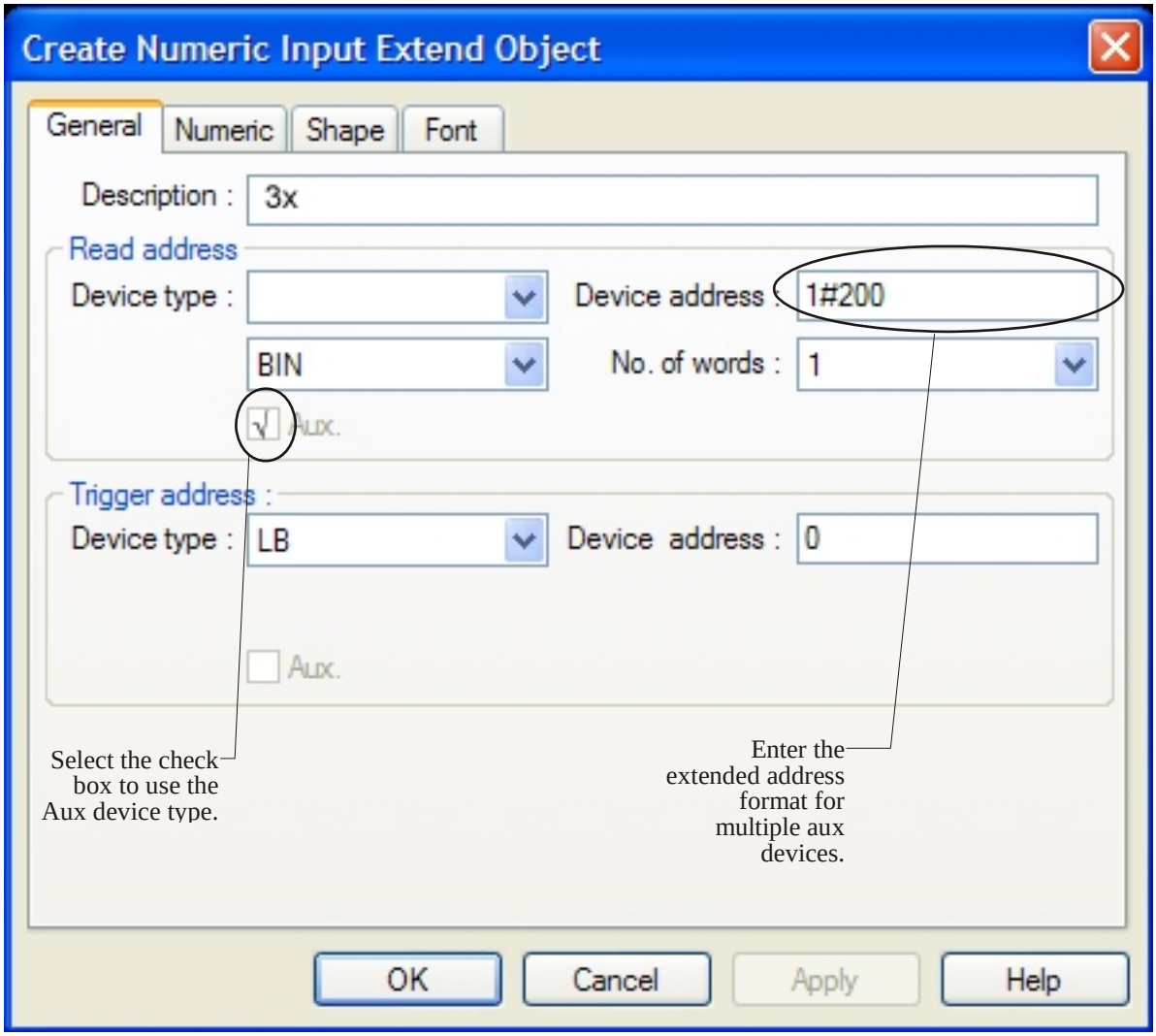
1. Set *Aux station*. This is used when the Aux device has a node or station identifier. The HMI needs the station number to initiate communications. Set as needed or leave at 0 if not used. Station numbers are 0-127 (use the range as appropriate for PLC type.) If the Extended address mode* is selected, the HMI will ignore the aux station setting.
2. Set *Aux timeout constant.* This setting determines how long (in seconds) the HMI will wait for a response from the Aux device. The range is from 0 to 127 seconds.
3. Set *Aux block pack.* This is used to determine how the unit will communicate (register polling) to the Aux device. By increasing this number, larger blocks of registers can be fetched from the Aux device by issuing one command for a block of registers instead of a command for each individual register. In some cases, this speeds the update of information on the display, but in other cases it may cause unwanted delays. Experiment with this number to optimize.

---

* ***Extended address mode:*** *When more than one aux device is connected on a RS485 bus, select the Extended address mode in the* ***Editor*** *tab of the* ***System Parameter Setting.*** *Then use the extended addressing format for all object addresses (i.e. 4x: 2#1 =Modbus address 40001 for node #2).*

To enter a read or write address for the Aux port

When Entering in data register types and address using the extended mode to be used by the Aux port device, select the Aux. check box first, then add the address in the extended mode format [node address]#[address].



Chapter 5 - Creating Windows

This section shows how to create windows using EasyBuilder. To better illustrate some of the examples, please open the sample project (MT506M.EPJ, MT506C.EPJ, MT506T.EPJ, MT508C.EPJ, MT508T.EPJ or MT510H.EPJ) included with the EZware-500 configuration software.

Windows Fundamentals

An operator interface terminal wouldn't be very useful if all of the information to be displayed could only be placed onto one screen. Therefore, most HMIs have multiple screens that you can use to display information. The Weintek MT5xx is capable of storing up to 1999 windows (actual limit is determined by memory requirements of each screen), giving you maximum flexibility in designing your operator interface. We prefer to call these screens 'windows' because they have several features not normally associated with screens:

- Windows can be created in any size. You can make the window full-sized so that it fits the entire area of the HMI display or you can create a window that partially covers the display.
- Windows can be overlaid on top of each other. A maximum of six windows can be displayed on the HMI at any given time. All data on each window displayed is updated continuously regardless of whether or not it is covered up by another window.
- Windows can be easily moved about the HMI display to allow portions of other windows to come into view.
- Windows can be minimized to an icon on the bottom of the HMI display. This feature allows the plant floor operator to minimize windows that contain data (such as numeric keys) which don't need to be seen all the time and yet must be readily available.

The MT5xx has three basic types of windows available for use: Base windows, a Common window, and a Fast Selection window. Base windows are the windows that you will most often use. The common window and fast selection window are two windows reserved for special functions. By the end of this chapter, you should be able to create these windows and use the many features available to them.

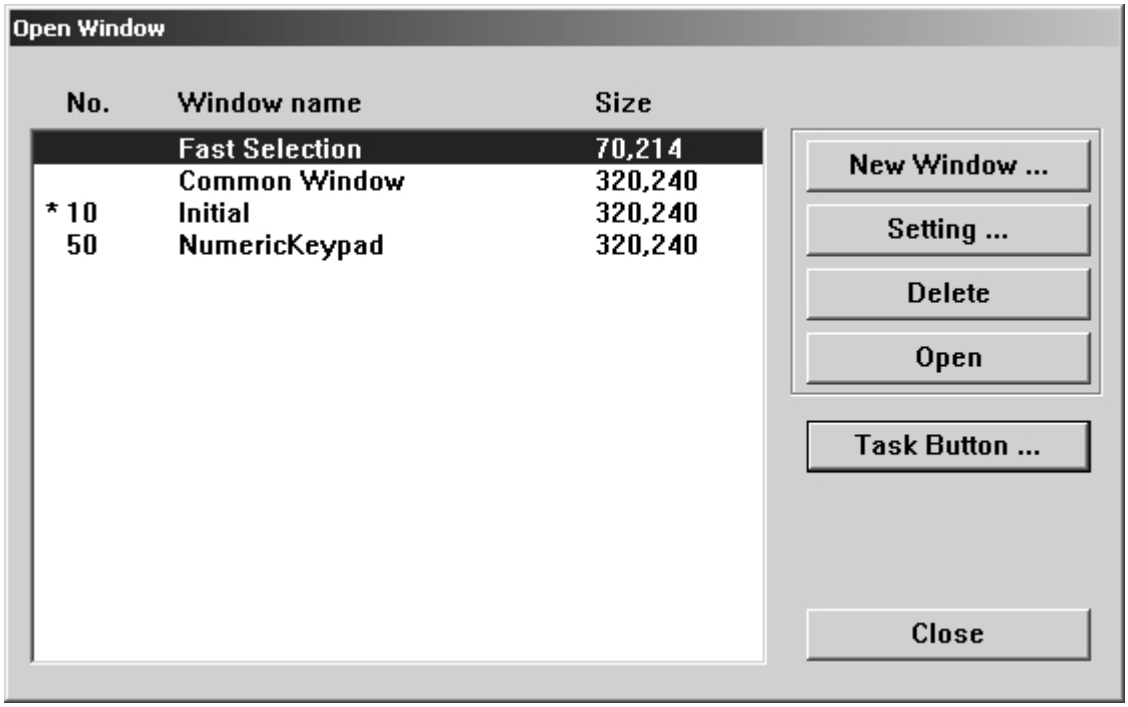
Opening a Window

To view the contents of a window in EasyBuilder, it must first be opened. When you create or open an existing project file only the initial window is opened. To view any other windows that have already been created, you must first open the window. This can be done using the Window Treebar (see Chapter 3, "Using EZware-500", Display Options) or by performing the following:

To open a window

1. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.
2. Click on the Window you wish to open. Then click **Open**. Note: you can also open the window by double-clicking the window.
3. The Open Window dialog box closes and the opened window is displayed in the EasyBuilder work area.

Let's look at the Open Window dialog box again using the sample project.



The Open Window dialog box lists all of the windows currently created for the project. You will notice that in the sample project, three windows have been created: 10, 11, and 12. The asterisk next to Window #10 indicates that this window has already been opened. Each window is listed with the Window name and size. The window name is the name that you assign to the window when it is created. The window size is shown for quick reference. Finally, you will notice that the Open Window dialog box is also used to create a new window, change any of the settings for a window, or delete a window. Press the **Close** command button to return to the main EasyBuilder screen.

By default, when a window is opened it will replace any window that was displayed in the work area of EasyBuilder. To switch between open windows, click the **Window** menu and select from the list of windows currently open. You can also *cascade* or *tile* the open windows to see the windows at the same time.

EasyBuilder requires more resources from your computer every time you open another window. When many windows are open, the performance of the computer may be affected. Therefore, you may wish to close some of the windows until you need to edit them.

▶ **To close a window**

1. You will notice three small icons located in the upper right hand corner of each window: the minimize icon , the maximize icon , and the close icon . To close a window, click the close icon associated with that window .

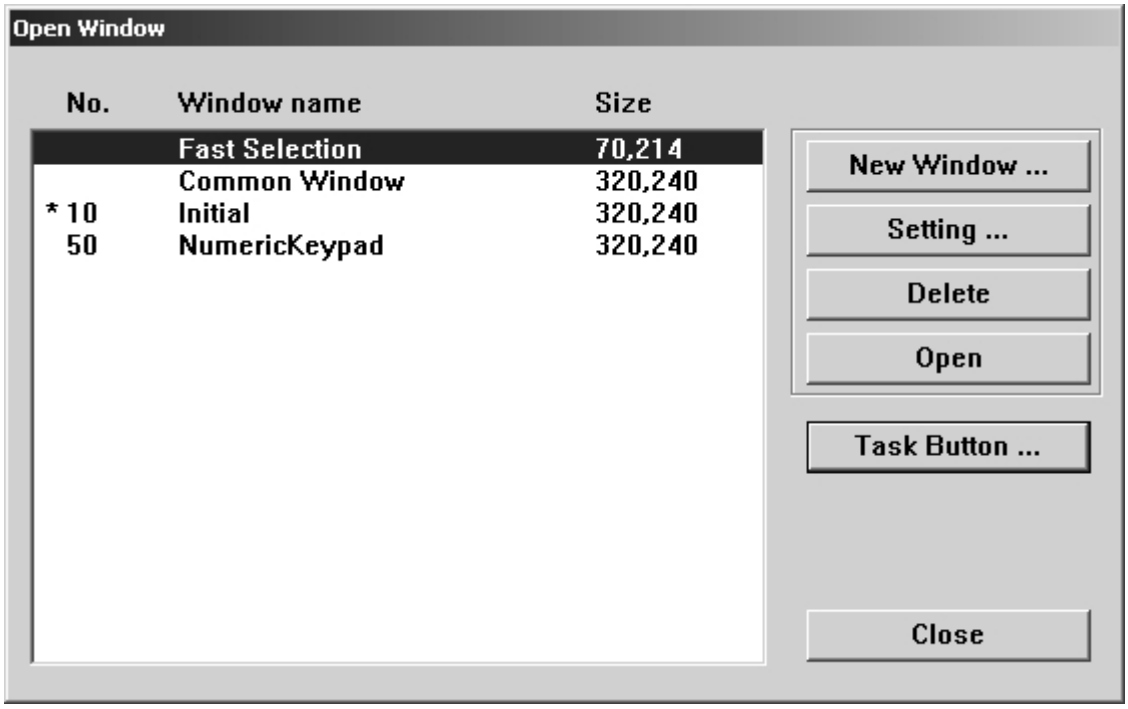
These icons show in the window if minimized. Otherwise, the icons are located on the rightmost end of the menu bar of EasyBuilder.

Creating a New Window

Whenever you create a new project, the initial window (Window #10), the fast selection window (Window #4), the common window (Window #6), and the numeric keypad window (Window #50) are automatically created. To create a new window, the following steps must be performed.

To create a new window

1. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.



2. Click **New Window**. The Select Window Style dialog box appears.

3. Click **Base Window.** The Fast Selection and Common Window buttons appear grayed out because those windows have already been created. The Window Setting dialog box appears.
4. Modify the window parameters, then press **OK**. The Open Window dialog box reappears.
5. If you wish to open the window you just created, click on the window and click **Open**. Otherwise, click **Close** to return to the main screen of EasyBuilder.

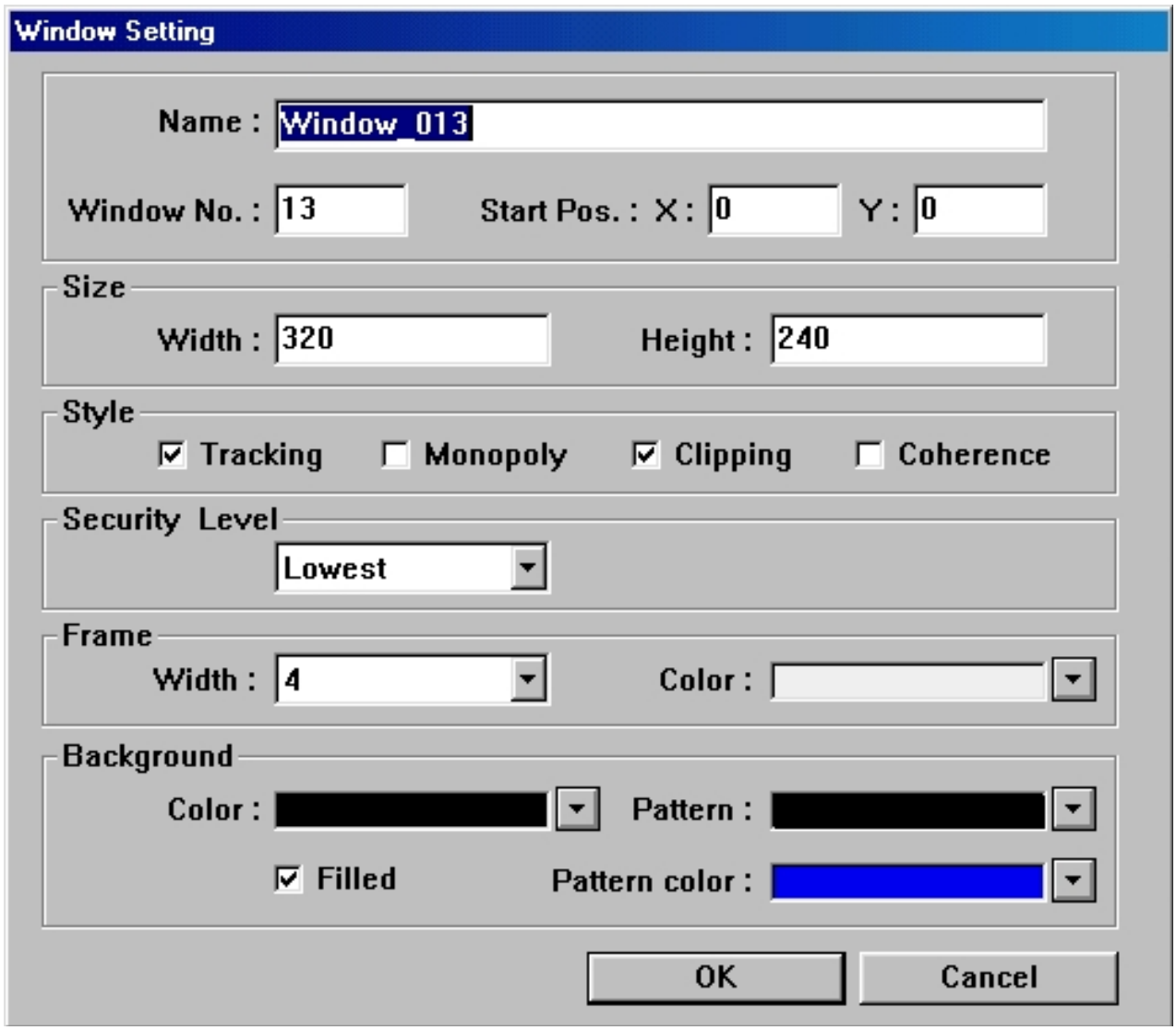
Window Settings

Let's look more closely at the parameters you can change when creating a new window.

When a window is initially created, the window settings can be edited to make changes; however, once a window has been created, the window number can not be changed. To change the window settings, highlight the desired window and click on the ***Setting...*** *button.*

Assigning a Window Name

The Name is a description box used to help you identify what the window is used for without having to actually open the window and look at the contents. Up to 50 characters can be entered into this field with space characters allowed.



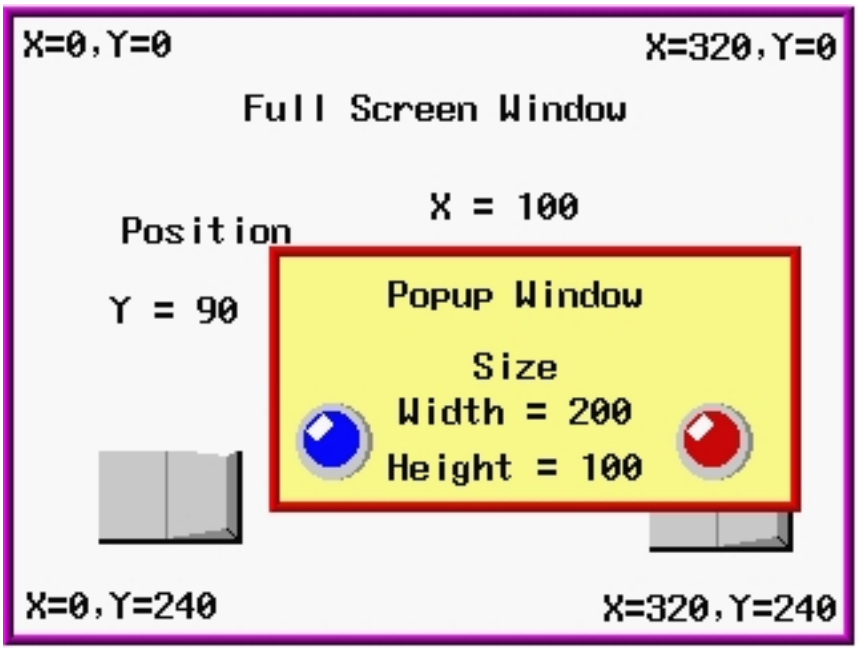
Assigning the Window Number

Although 1999 windows are available on the MT5xx, two are specifically reserved for the Common Window and t Fast Selection window and another eight are reserved for internal use. Therefore, you can assign #10-255 to any window you create. When you initially create a window, EasyBuilder will automatically assign the lowest available number to the window. However, you can assign any number within the allowed range. In this manner, you can group windows together that may share some common traits.

Assigning a Position

This is the starting position that the window goes to when it is initially called onto the HMI screen. The starting position is labeled Start Pos: in the Window Setting Dialog box. The X and Y positions refer to the pixel location of the HMI display at which the upper left hand corner of the window is to be displayed.

The default setting is X=0 and Y=0 which is the upper left hand corner of the HMI display. The X-axis refers to the horizontal location and the Y-axis refers to the vertical location. The MT506C has a 320 x 240 (MT506T is 320 x 234) pixel display, so the ranges are X=0-319 and Y=0-239. The MT508/550 has a 640 x 480 pixel display, so th ranges are X=0-639 and Y=0-479.



Assigning Size of Window

You can vary the size of a new window to create full screen or popup windows. Popup windows are most often used to display data that does not need to be on the HMI display all the time. For example, you might want to configure a numeric keypad in one popup window that can be minimized to an icon when not needed. To use the numeric keypad, just maximize the popup window. The allowable range for the width is 100-320 (640 for the MT508/550) pixels. The allowable range for the height is 100-240 (234 for the MT506T, 480 for the MT508/550) pixels.

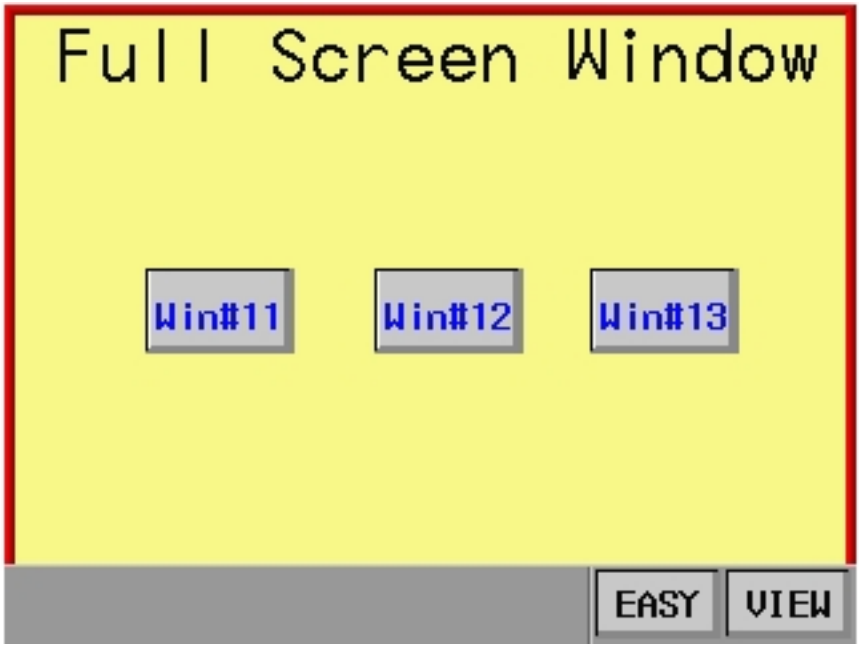
Window Styles

There are several options that can be assigned to each window created that affect how it operates with other popup windows.

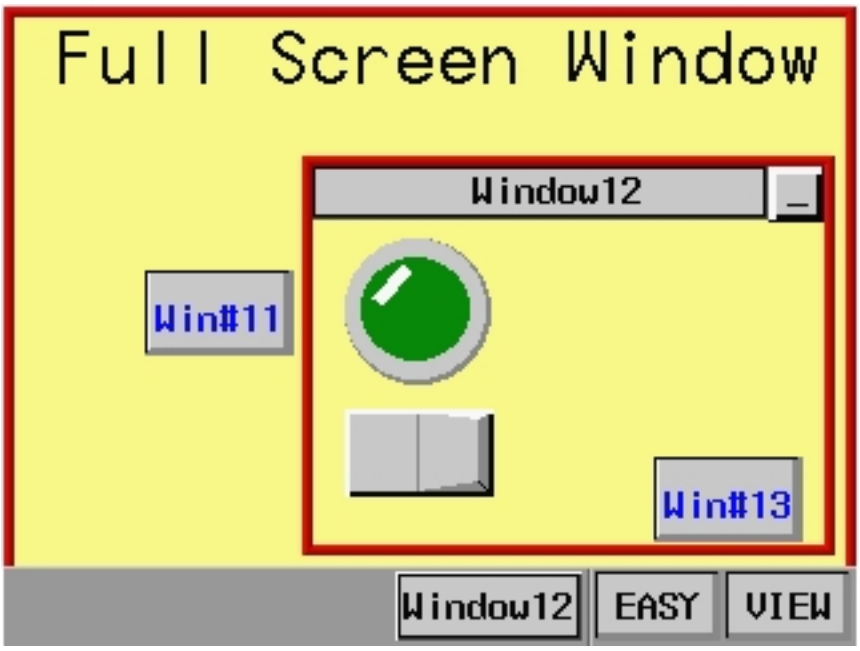
Tracking Feature

The tracking feature causes any popup window that was called from another popup window to move when the first popup window is moved. Think of a ‘parent/child’ relationship in which the calling popup window is the parent and the called popup window is the child. Any place that the parent window moves, the child window follows. This includes maximizing and minimizing the windows into an icon. If tracking is enabled and the parent window is

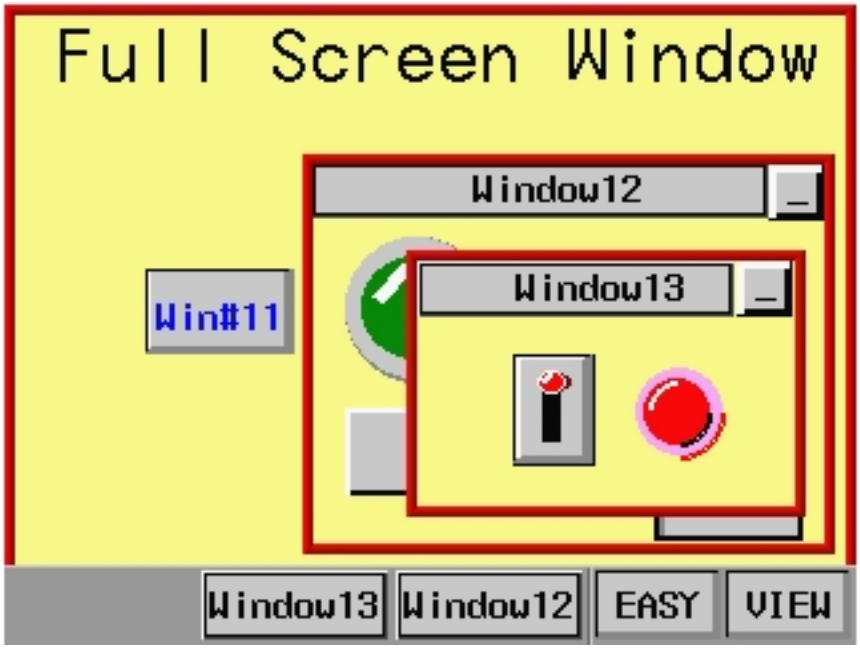
minimized to an icon, then the child window is automatically minimized at the same time. For example, suppose that the HMI displays the following:



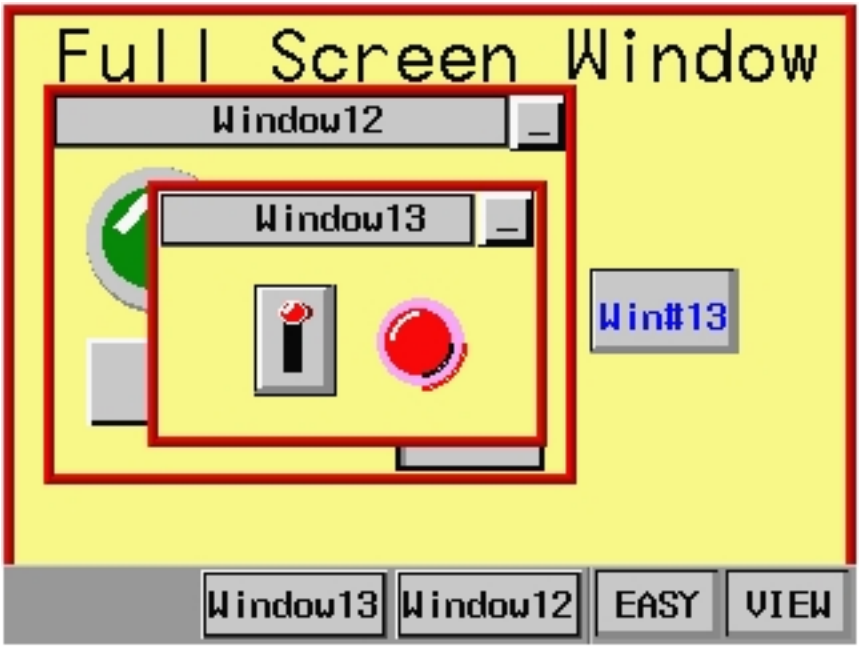
Then popup window #12 is displayed:



We then use the function key on popup window #12 to display popup window #13.



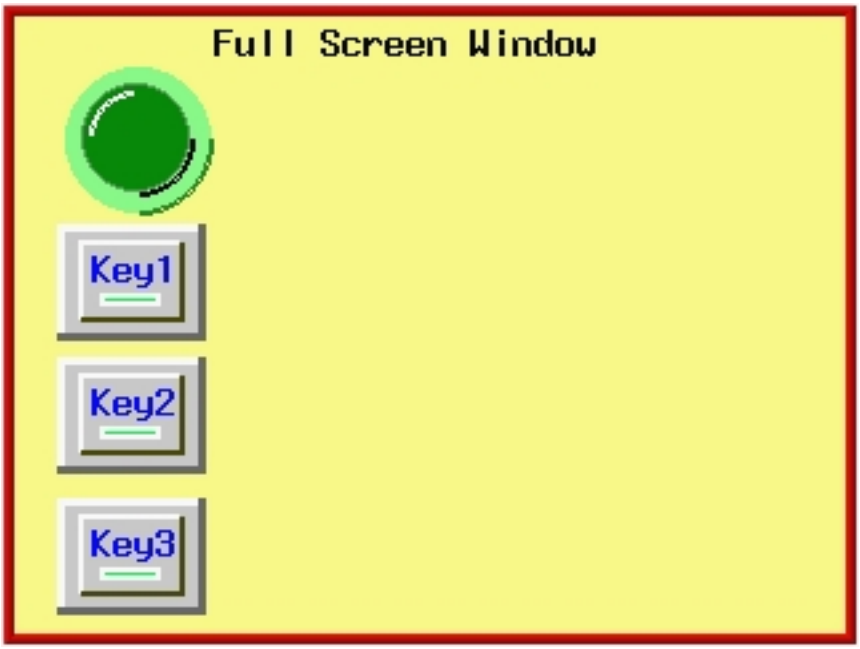
If window #13 has the tracking feature enabled, then moving window #12 (the parent) will cause window #13 (the child) to move as well. This is because window #13 was 'called' from an object on window #12.



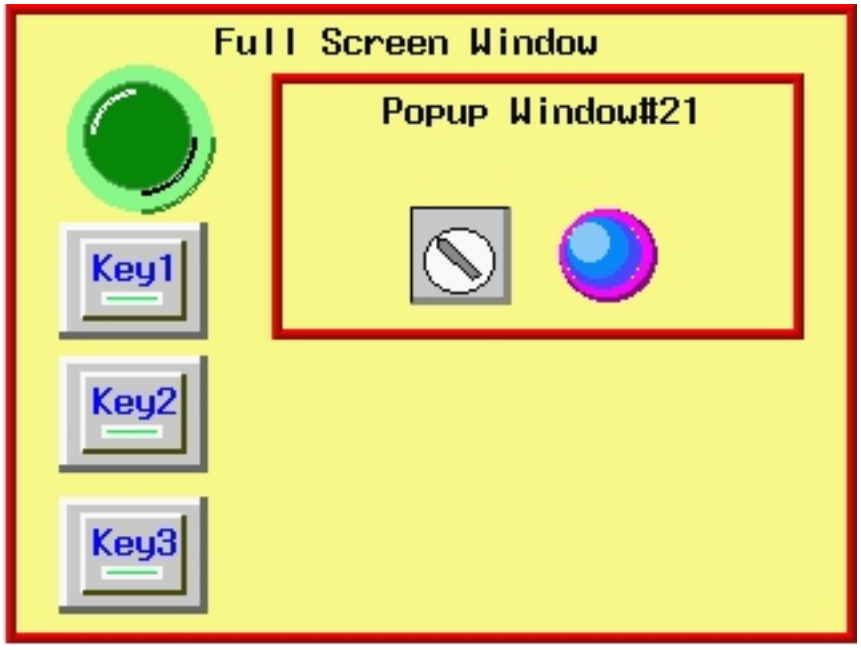
The tracking feature is often used in instances in which you want to make it very clear that the information on one window (the child) always pertains to another window (the parent).

Monopoly Feature

The monopoly feature is used to ‘monopolize’ all touch screen action that can occur on the HMI screen. For instance, suppose the HMI display is currently showing a full screen window with several touch screen objects.

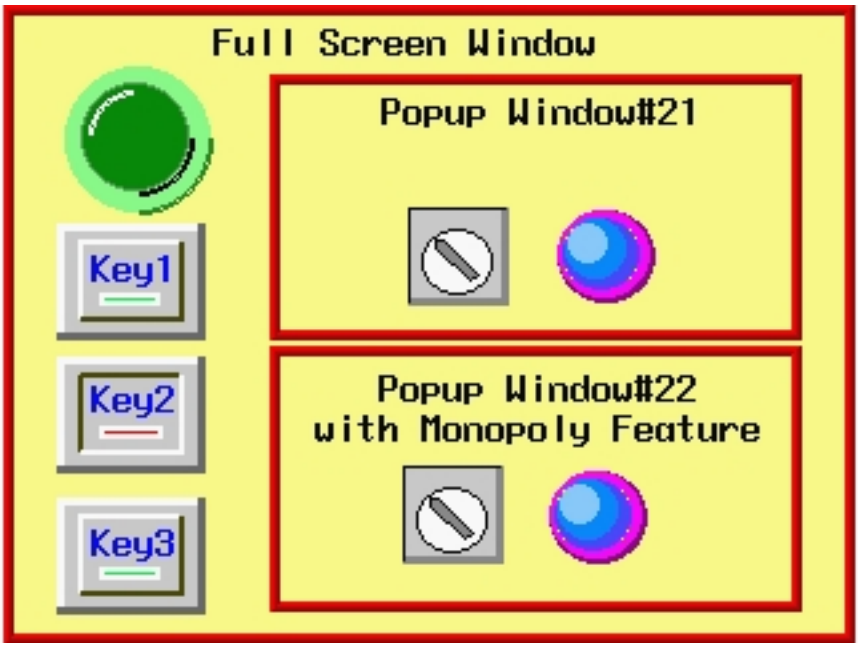


If a popup window is displayed which *does not* have the monopoly feature enabled, the HMI operator is able to press any function keys on the popup window or the full screen window since objects on both screens are active.



In this example, the HMI operator is able to press Key1-3 on the full screen window or the switch on Popup

Window#21. If another popup window is displayed that does have the monopoly feature enabled, then Key1-3 on the full screen window will not respond when pressed.



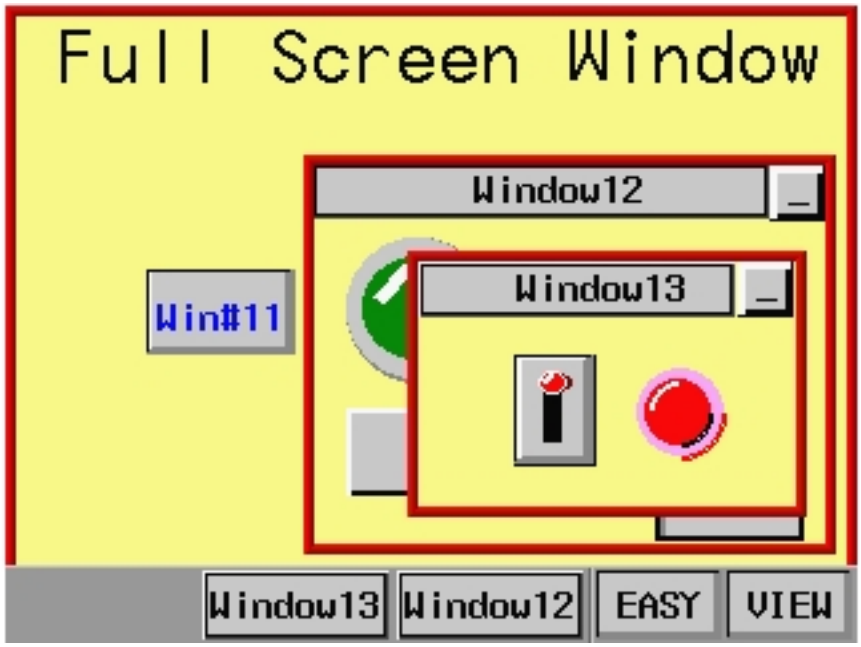
Please note that the monopoly feature only disables touch screen objects on the *full screen window*. The switch on Popup Window#21 can still be activated, as can any popup window displayed on the full screen.

The monopoly feature can be used to display a popup window with some action that the HMI operator must perform before being allowed to do some other action on the HMI. For example, you might construct a dialog box that asks the HMI operator if some step in the control process has been performed. The dialog box would have **Yes** and **No** function key options which, with the monopoly feature enabled, the HMI operator must press before continuing.

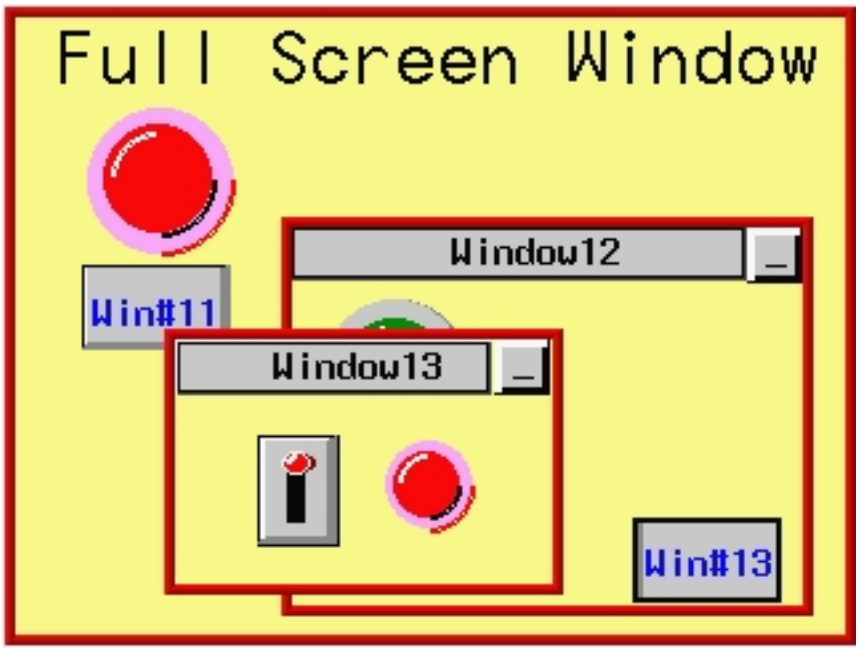
Clipping Feature

With clipping enabled, any portion of a child window that lies outside the boundary of the parent window is 'clipped' or not displayed on the HMI screen. The tracking feature must be enabled when the clipping option is

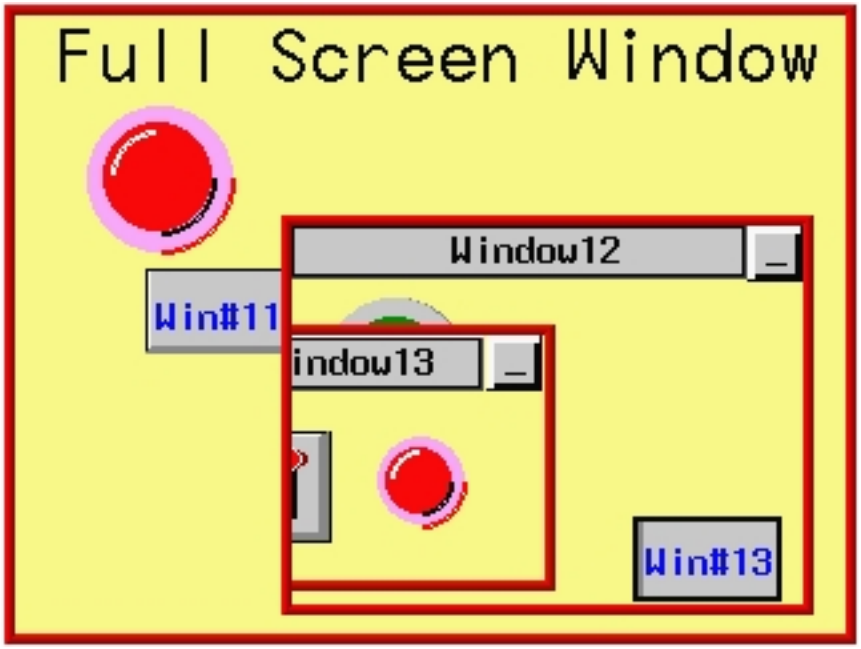
used. For this example, let's look at an HMI screen in which Popup Window#13 is a child window that was called by an object in Popup Window#12, the parent window.



Without clipping, if we move Window#13 so that a portion of the window is out of the area covered by Window#12, then the entire area of Window#13 is still shown on the HMI screen.



With clipping enabled on Window#13, the same move would cut off part of Window#13.



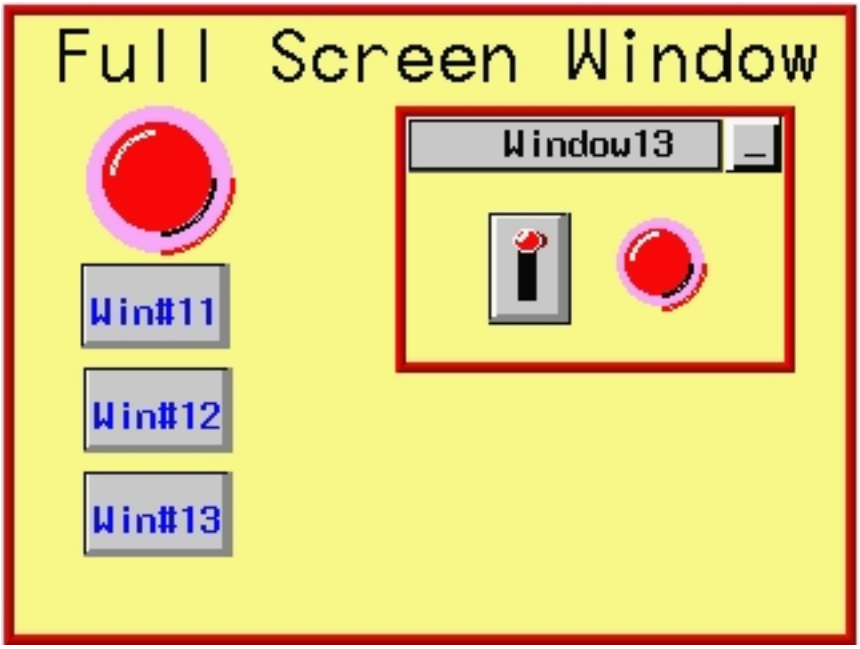
Like the tracking feature, clipping is most often used to reinforce the notion that one window is tied to the operation of another window.

Coherence Feature

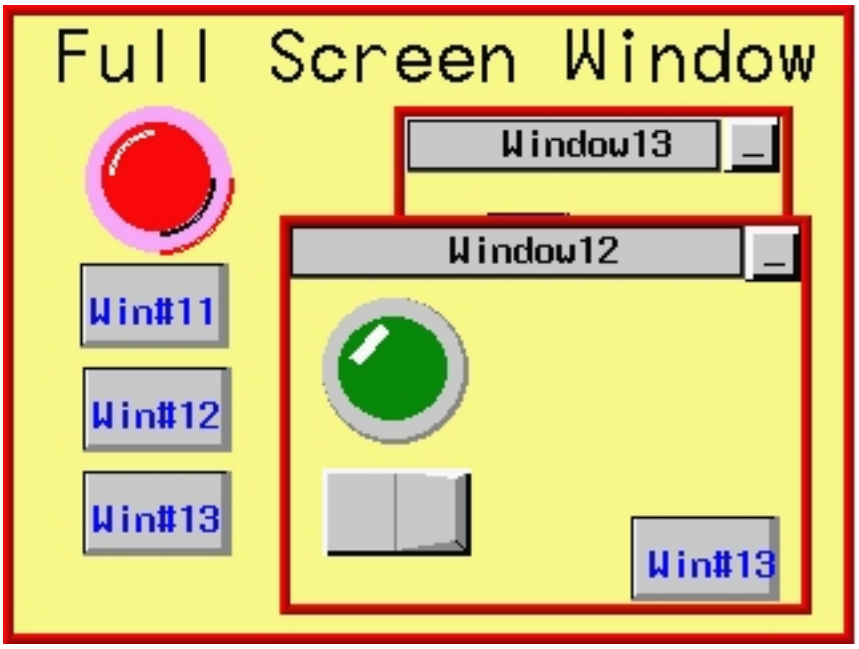
Normally, when a popup window is displayed on the HMI screen, it covers all other windows that are currently displayed on screen. The coherence feature causes the popup window to be displayed *underneath* all other windows currently displayed. In other words, the coherence feature makes that window subordinate to all other windows. When the coherence feature is applied to a popup window, you will see the entire popup window only if there are no other popup windows taking up the same area.

The coherence feature only works if the popup window that has this feature enabled, is called from a full screen base window. If the popup window (child) is called from another popup window (parent), then the child window will appear above the parent even with the coherence feature enabled.

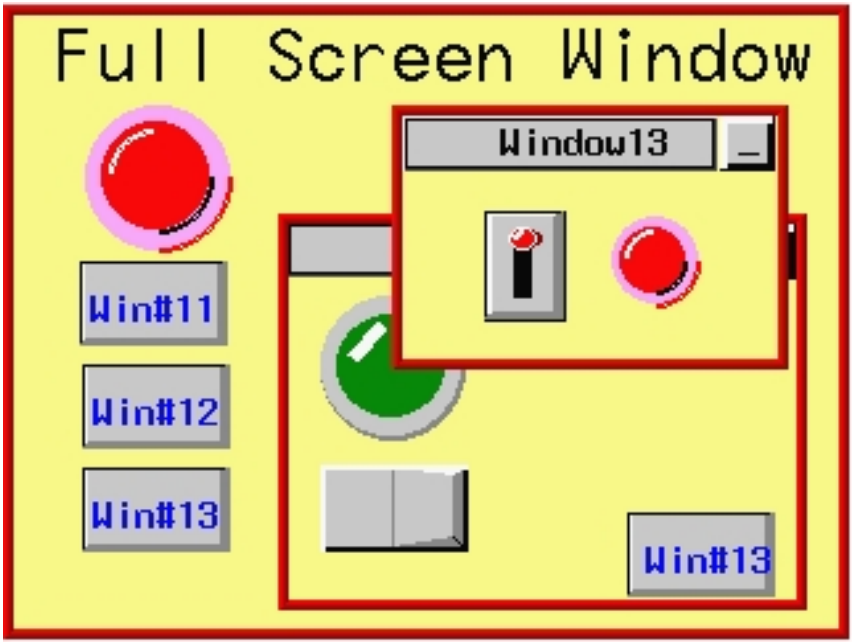
For example, suppose that you have a full screen that calls up Popup Window #13.



If the full screen calls a popup window (with coherence disabled), it displays on the HMI screen over any other popup windows that might occupy the same area. In this example, the full screen calls up Popup Window #12.



However, if Popup Window #12 had the coherence feature enabled, a different picture would have occurred.



Although Popup Window #12 was displayed after Popup Window#13, it appears behind it. Even if the HMI operator 'activates' Window #12 by touching some object on that window, it will remain behind all other popup windows displayed.

If two popup windows are displayed with the coherence feature enabled on both, then the popup window that is displayed first will be the dominant window.

Coherence is most often used on popup windows that take a large area of the HMI display. When they are displayed, they often completely cover smaller popup windows that are already on display, making it difficult to access the covered popup windows.

Setting the Window Security Level

EasyBuilder provides the ability to restrict access to window screens so that only authorized personnel can view them. Three security levels are available: Level 0 (Lowest), Level 1 (Middle), and Level 2 (Highest).

You must perform the following three steps to use the Window Security Level feature:

1. Assign passwords to the three levels of security access.
2. Assign a security level to each of the window screens.
3. Create a Numeric Input Object that allows you to change the current access level.

To assign Window Security Level passwords

1. From the **Edit** menu, select **System Parameters**. The Set System Parameters dialog box appears.
2. Click the **Security** tab to display the Security form.
3. To enable the Window Security Level feature, check the **Security Control** box.

4. The **Password** frame box appears showing the current passwords for the three security levels.
5. Each password can have a value of 0 (indicating no password is assigned to that level) or 1 to 4,294,967,295. After entering a password for each level, click **OK** to go back to the main screen of EasyBuilder.

These passwords must be entered by the HMI operator to gain access to any window screens that are assigned a security level.

To assign a security level to window screens

1. All window screens are initially configured with the lowest level security access allowing total accessibility. To change the level of access, you must change the window screen's security level in the window settings dialog box.
2. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.
3. Click on the window you wish to modify, and then click the **Setting** command button. The Window Setting dialog box appears.
4. In the Window Setting form, select **Lowest**, **Middle**, or **Highest** level from the **Security Level** drop-down list box. Click **OK** to return to the Open Window dialog box.
5. Click **Close** to return to the main screen of EasyBuilder.

To create the ability to change window security level

1. Create a window(s) that contains a Numeric Input Object. For more information on creating a Numeric Input Object, see "Representing PLC Data Registers" section in Chapter 6.
2. From the **Parts** menu, click **Numeric Input**. The Create Numeric Input Object dialog box appears.
3. In the **Read address** frame, select local word **LW9040**. LW9040 is a reserved internal data register of the HMI that is used to change the window security access level.
4. Select **BIN** (binary) format. Select **No. of words:** as 2.
5. Click the **Numeric** tab to display the Numeric form.
6. In the **Display** attribute box, select **Decimal** display or, if you wish to prevent others from seeing the password code, select **Mask** display.
7. Then create a numeric keypad to allow entering numbers into the Numeric Input Object, (see Chapter 7 – Using and Creating Keypads).

Two additional local word registers are used to support the **Window Security Level** feature:

- LW9042 is a 16-bit Read Only register which always shows the current window security level: 0, 1, or 2.
- LW9043 is a 16-bit Read/Write register that quickly 'forces' the window security level to a lower level by entering the level number. For instance, if the current security level is Level 2 (the highest security level), you can move quickly from this level to Level 1 or Level 0 by entering the number 1
  or 0 respectively into this register- no password is required. This feature enables someone who has the highest security level to switch the HMI to a lower level without having to memorize all three passwords.

Assigning Underlay Windows

EasyBuilder allows you to assign up to three underlay windows to any base window that you are creating. First, create the underlay window; then select it from the pull-down list box when you change the settings of the base window.

How to Display Underlay Windows

When you have common objects to display on multiple screens, use the Underlay Window feature to store these on underlay windows attached to a base window. This decreases the amount of memory required for the project.

which are called Underlay #1, Underlay #2, and Underlay #3. Objects on the underlay windows cannot be edited from the base window on which they are displayed.

A pop-up window cannot display any underlay windows.

To assign underlay windows to a base window, you must first create the underlay windows. Underlay windows are created the same way base windows are created. Then you can assign the underlay windows to the base window.

To assign underlay windows to a base window

1. From the Window menu, select Open Window. The Open Window dialog box appears.
2. Highlight the base window you wish to use.
3. Click the Setting command button. The Window Setting dialog box appears.
4. Click the pull-down list box for each Underlay Window to assign the underlay window screens to the base window.
5. Click **OK**. Click **Close** on the Window Setting dialog box.

Rules That Apply to Underlay Windows

Please note the rules that apply when you use the underlay window feature:

- Only the objects of an underlay window are displayed on the base window. All background information about the underlay window (i.e. background color, frame color and size, etc.) are not displayed on the base window.
- Active objects can be 'overlaid' on top of each other over two or more underlay windows. For example, a Set Bit object that is configured to set an HMI local bit LB0 is located at X=20, Y=50 for Underlay Window #1. Another Set Bit object that controls LB1 is placed on Underlay Window #2 at the same location. When the Base Window displays these two windows, both objects will be active so that when they are pressed both LB0 and LB1 will be set.
- Static objects may be overlaid using underlay windows. However, any static object that is on the Base Window has the highest precedence and will be displayed over any static objects that are on the underlay windows. Active objects have higher precedence over static objects.
- Popup windows cannot display underlay windows.
- Underlay windows are always positioned from the top left corner of the screen. Any position settings you make when creating the underlay window are ignored when the window is displayed by a Base Window as an underlay window.

For more information and an example of how to use underlay windows, consult the EasyBuilder Help files.

Creating a Frame

EasyBuilder provides the option of having a frame around any window that you create. You can select from seven sizes: 4, 6, 8, 10, 12, 14, or 16 pixels wide. The default setting is no frame. If a frame is selected, you can select one of 16 colors when using the MT506C, MT506T, MT508C, MT508T or MT510H color HMIs or one of four shades for the MT506M monochrome HMI.

Window Background

You can also select a different background color for each window created. The default setting is black.

To select a different background color

1. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.
2. Click on the window that you wish to change the background color. Click **Setting**. The Window Setting dialog box appears.
3. In the Background frame section, check the **Filled** box.
4. Click the pull-down arrow of the **Color** box. The Color dialog box appears.

5. Select from one of the 16 basic colors or add 16 new colors from the customized list of colors. To create a customized color, click on one of the boxes in the Customized color box, then click **Customize color**.
6. The color table dialog box appears allowing you to select one of the 256 colors available.
7. Click **OK** in the Color dialog box to go back to the Window Setting dialog box. Click **Close** to return to the EasyBuilder main screen.

Instead of using a solid background color, you can select a pattern that is displayed in the background of a window.

To select a background pattern

1. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.
2. Click on the window that you wish to change the background pattern. Click **Setting**. The Window Setting dialog box appears.
3. In the Background frame section, check the **Filled** box.
4. Click the pull-down arrow of the **Color** box. The Color dialog box appears.
5. Select from one of the 16 basic colors or add 16 new colors from the customized list of colors. To create a customized color, click on one of the boxes in the Customized color box, then click **Customize color**.
6. The color table dialog box appears allowing you to select one of the 256 colors available.
7. Click **OK** in the Color dialog box to go back to the Window Setting dialog box.
8. Click the pull-down arrow of the **Pattern Color** box and select a color. Note: to use the background pattern, the color selected for Pattern Color must be different then the color selected for the solid color.
9. Once a pattern color is selected, click **OK** in the Color dialog box to go back to the Window Setting dialog box.
10. Click the pull-down arrow of the **Pattern** box. The Pattern Style dialog box appears.
11. Select the pattern you wish to use, then click **OK**. The Window Setting dialog box reappears.
12. Click **OK** in the Color dialog box to go back to the Window Setting dialog box. Click **Close** to return to the EasyBuilder main screen.

Deleting a Window

Before any window can be deleted from a project, the window must be closed. To close a window, see the section earlier in this chapter on opening windows.

To delete a window

1. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.
2. Highlight the window that you wish to delete.
3. Click the **Delete** command. A dialog box appears asking if you want to delete this window.
4. Click **Yes**. The dialog box disappears and the selected window is deleted.
5. Click **Close** in the Open Window dialog box to go back to the EasyBuilder main screen.

Using Base Windows

Of the three types of windows, base windows are the most commonly used. A base window is used to create a full screen window or a popup (partially sized) window. You may have up to six popup windows on the screen at any moment in time. Base windows can be minimized to an icon on the task bar. Base windows can also be moved
about on the HMI display and can overlap each other. The number of objects that can be placed onto each base

How to Display Base Windows

Base windows can be displayed on the HMI screen by using a function key to display the window or by using the PLC to call up the window.

Using a Function Key

The Function Key Object is a graphics touch screen object that you place onto a window to perform an action. Function keys have many purposes which are discussed in later sections, but two actions that a function key can perform are:

- Calling a full screen window
- Calling a popup window

Calling a full screen window

If a function key object is created to call a full screen window, the window that is displayed replaces all other windows that are on display regardless of how many are open. Therefore, think of calling a full screen window as performing two actions: closing any open windows and displaying a full screen window.

To call a full screen window using a function key object

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box appears.
2. In the General Tab, click Change Window.
3. In the same frame, enter the **Window No**. you want to call.
4. In the **Shape** tab, click on the **shape** or **bitmap** checkbox, and then click on the **Shape library** or **Bitmap library.** Select the shape or bitmap you wish to use to represent the function key.
5. In the **Label** tab, select a label for the function key.
6. Click **OK** to return to the main screen of EasyBuilder.
7. Place the function key object where you want it on the window you are editing.

You must select a window that is full size (320 x 240 for the MT506C/M, 320 x 234 for the MT506T, 640 x 480 for the MT508/550) pixels when using the Change Window function.

Calling a popup window

If a function key object is created to call a popup window, the window that is displayed is generally overlaid over all other windows that are on display. Therefore, think of calling a popup window as opening another active window for display. Note that the maximum number of popup windows displayed at any one time is six.

To call a popup window using a function key object

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box appears.
2. In the **General** Tab, click **Popup Window**.
3. In the same frame, enter the **Window No**. you want to call.
4. In the **Shape** tab, click on the **shape** or **bitmap** checkbox, then click on the **Shape library** or **Bitmap library.** Select the shape or bitmap you wish to use to represent the function key.
5. In the **Label** tab, select a label for the function key.
6. Click **OK** to return to the main screen of EasyBuilder.
7. Place the function key object where you want it on the window you are editing.

You must select a window that is less than full size (320 x 240 for the MT506C/M, 320 x 234 for the MT506T,, 640 x 480 for the MT508/550) pixels when using the Popup Window function.

To display a full screen or popup window, press the touch screen where the function key object is located. The window displays in the location on the HMI display as determined by the X and Y position in the Window Settings box for that window.

For more information on using the Function Key object, see the chapter on "Representing Data with Graphics Objects".

Using the PLC to display a base window

Three objects are used by the PLC to call or display a base window. The PLC Control Object is used by the PLC to display full screen windows. The Direct and Indirect Window Objects are used by the PLC to display popup windows.

The PLC Control Object

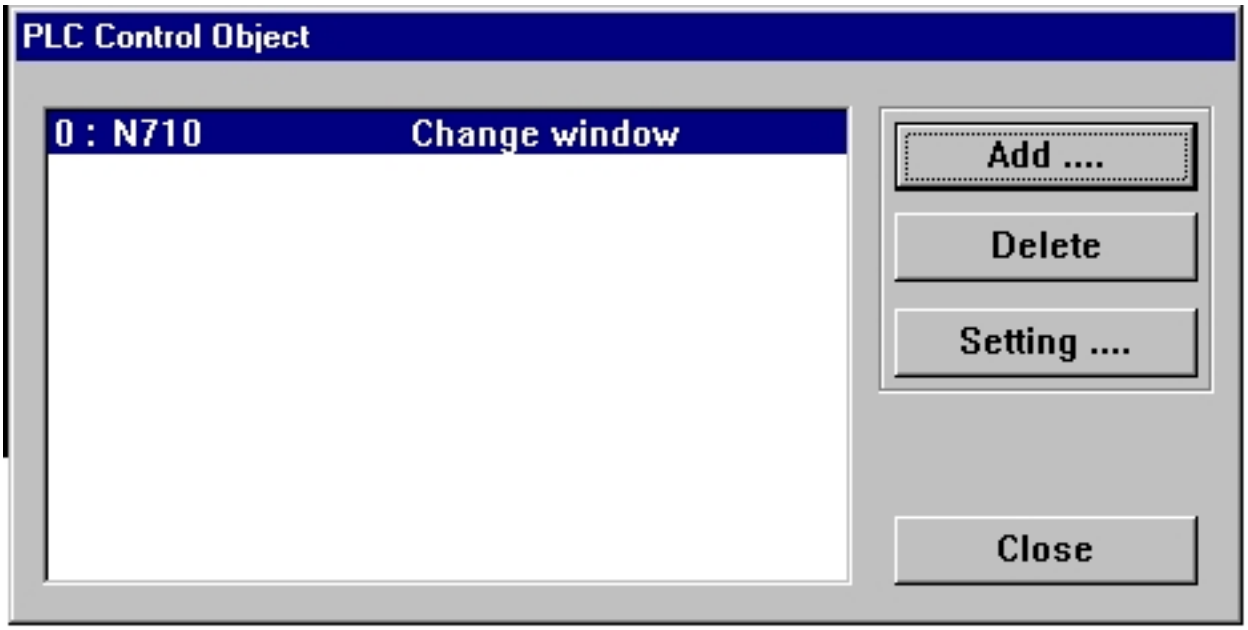
The PLC can display a full screen window by using the PLC Control Object. This object allows the HMI to continuously scan a PLC register to display a full screen window, which corresponds to the number in the PLC register.

Once the HMI displays a full screen that is requested by the PLC Control Object, the HMI will automatically write the number of the requested screen to the next consecutive register. For example, if you assign internal data register LW10 to a PLC Control Object/Change Window and the number 3 is placed into this register, then the HMI will display Screen #3. Finally, it will put the number 3 into LW11. This allows the PLC to confirm that the Screen has been properly displayed by the HMI.

You can create as many PLC Control Objects as you need- each object is universal and is not dependent upon which HMI screen is currently on display.

To call a full screen window using the PLC Control object

1. From the **Parts** menu, select **PLC Control**. The PLC Control Object dialog box appears.



2. Click **Add**. The PLC Control Object's Attribute dialog box appears.

3. Select the **Device type** and **Device Address** for the actual PLC address you wish to monitor.

The HMI will write the number of the newly-displayed window to the next consecutive address after the address specified in this step.

4. Select either **BIN** (binary) or **BCD** format.
5. Click **OK** to return to the PLC Control Object dialog box.
6. You should see a new entry that lists the PLC address that is monitored by the HMI.

The PLC must only enter window numbers that represent full size (MT506C: 320 x 240, MT506T: 320 x 324, MT508/550: 640 x 480) windows when using the PLC Control Object function. The HMI automatically closes any open windows before the window called by the PLC Control Object is displayed.

The Direct and Indirect Window Objects

If you want the PLC to call up a popup window that is to be overlaid on top of other windows already open, then there are two methods used to do this:

- Direct Window Object
- Indirect Window Object

Unlike the PLC Control Object, the Direct and Indirect Window Objects are only active for the windows that they are placed into.

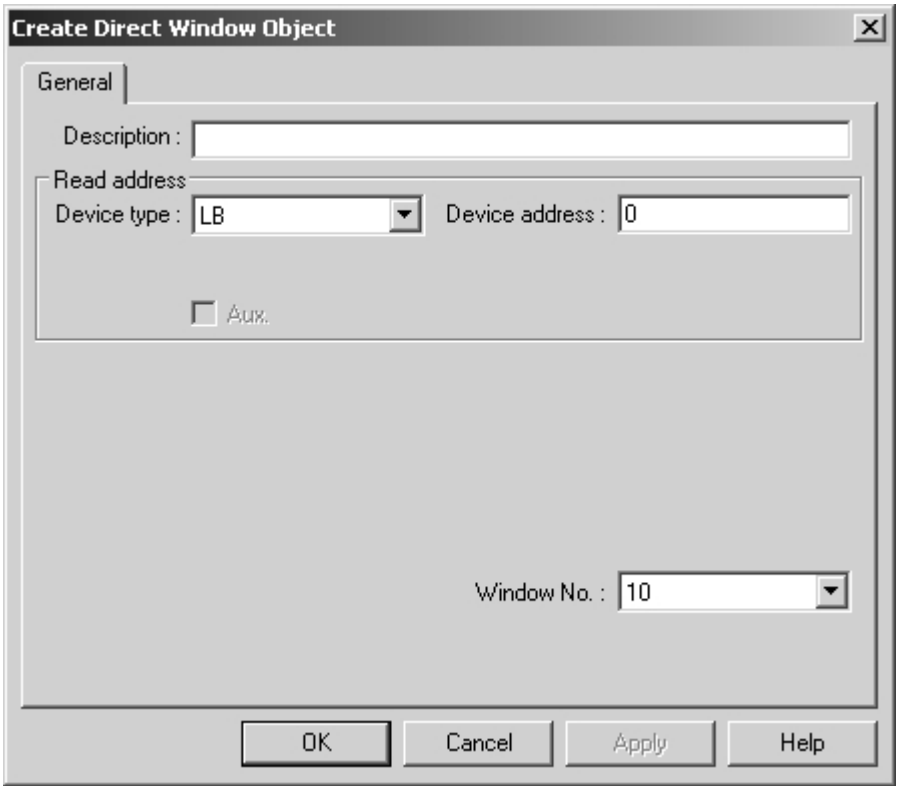
The Direct Window Object is used to display a popup window using a PLC coil. The HMI continuously reads the value of the PLC coil to determine if it is set to 1. If so, then the predefined popup window is displayed.

The Indirect Window Object is used to display a popup window using a PLC data register. Similar to the PLC Control Object, the HMI continuously monitors the selected PLC data register. It will display any popup window that corresponds to the number placed into the PLC data register by the PLC.

The PLC must reference a window that is less than full size (MT506: 320 x 240, MT508/550: 640 x 480) when using the Direct Window and Indirect Window objects.

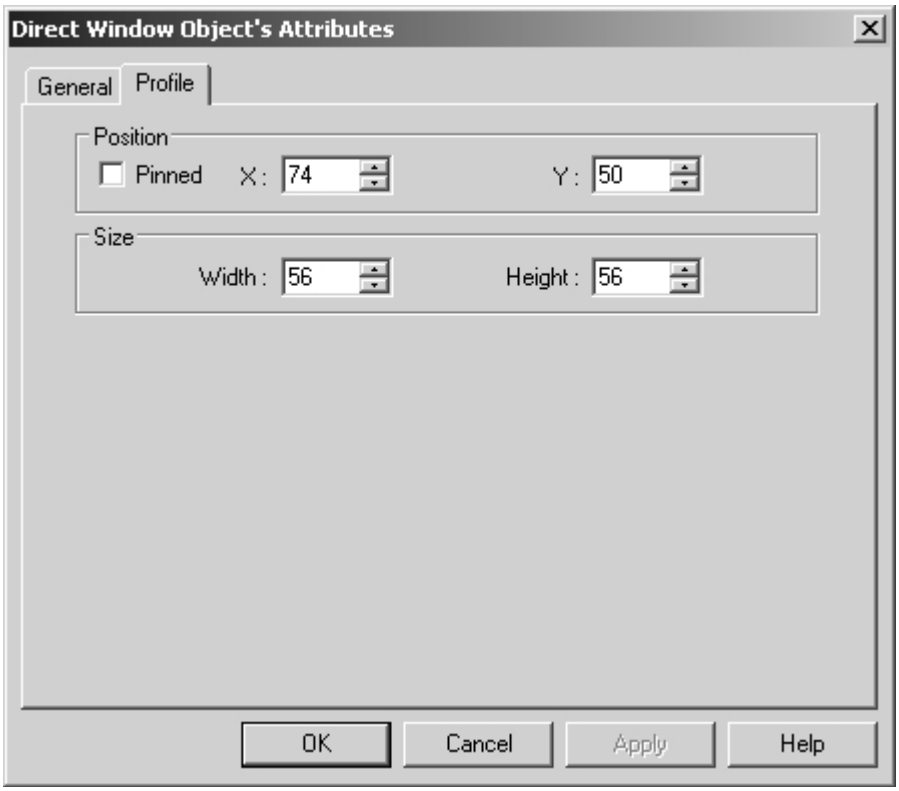
To call a popup window using the Direct Window Object 

1. From the **Parts** menu, select **Direct Window**. The Create Direct Window Object dialog box appears.



2. In the **General** Tab, select the PLC address according to **Device Type** and **Device Address**.
3. In the same tab, enter the **Window No**. you want to call.
4. Click **OK** to return to the main screen of EasyBuilder. The mouse cursor will have a square object that represents the size of the popup window.
5. Place the Direct Window object where you want the popup window to appear.

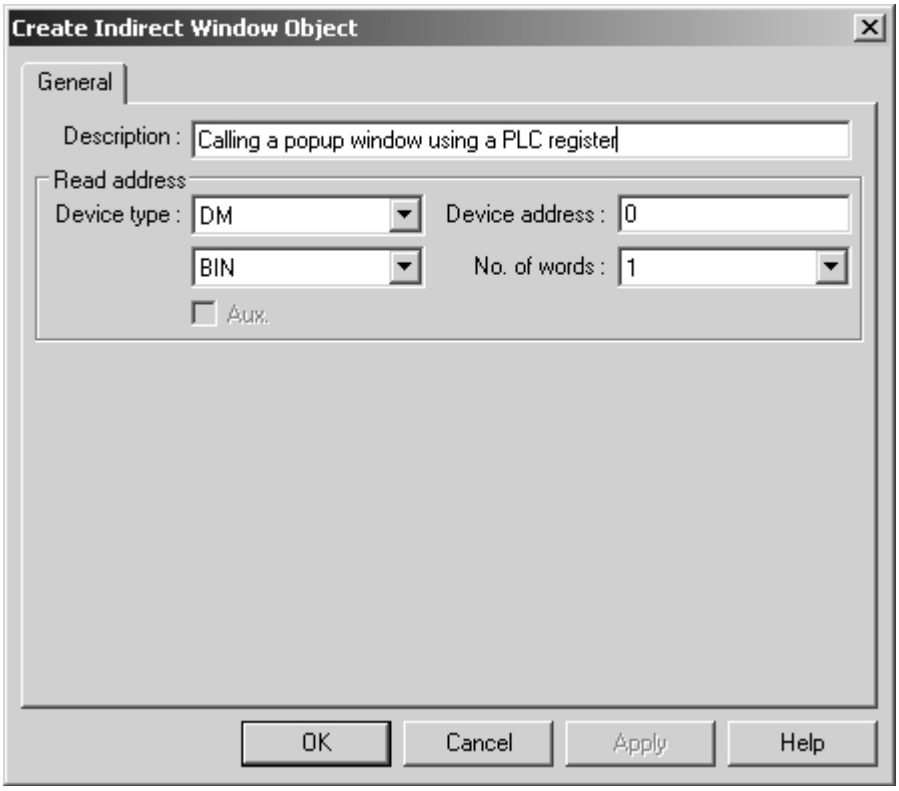
6. You can resize the object after you have placed it or double-click on the object to display its attributes.



7. Click the **Profile** tab to change the position or the size of the Direct Window Object. Note that the size *does not refer* to the size of the popup window since the size of the popup window is determined in the Window Settings dialog box. This size refers to the area of the Direct Window Object which is analogous to a box in which the popup window is to be placed. If the size is too small, then the popup window will appear clipped when displayed. Similarly, if the size of the Direct Window Object is larger than the actual size of the popup window, the excess space will be blank.

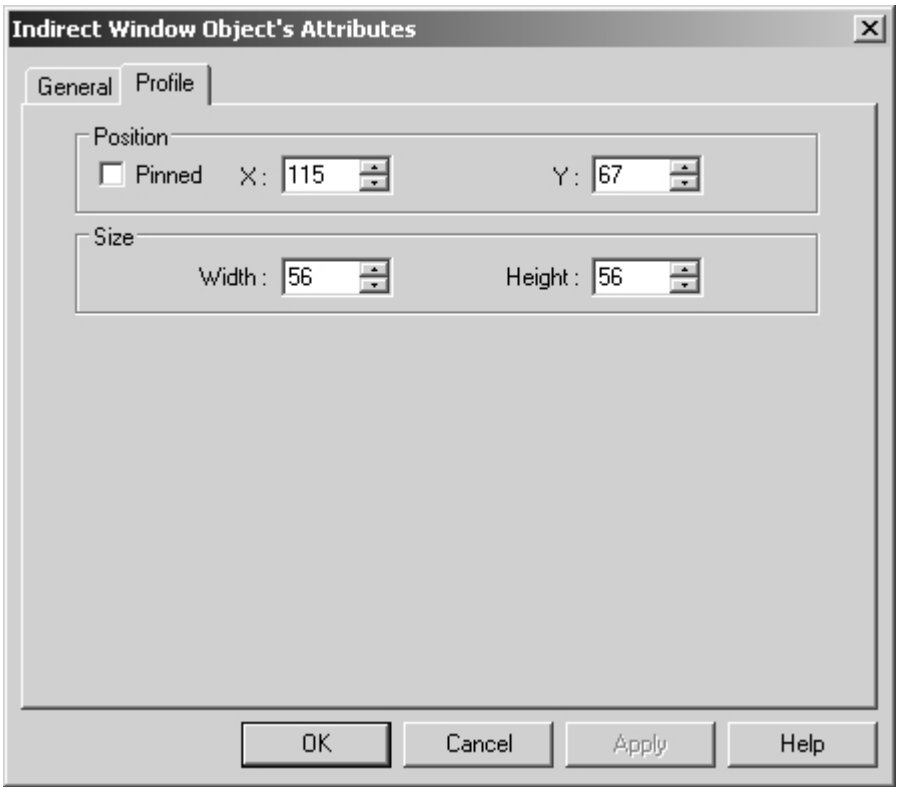
To call a popup window using the Indirect Window Object 

1. From the **Parts** menu, select **Indirect Window**. The Create Indirect Window Object dialog box appears.



2. In the **General** Tab, select the PLC address according to **Device Type** and **Device Address**.
3. Select **BIN** or **BCD** format.
4. **No. of Words** is currently set to 1 and cannot be changed. This represents a 16-bit PLC data register.
5. Click **OK** to return to the main screen of EasyBuilder. The mouse cursor will have a square object that represents the size of the popup window.
6. Place the Indirect Window object where you want the popup window to appear.

7. You can resize the object after you have placed it or double-click on the object to display its attributes.



8. Click the **Profile** tab to change the position or the size of the Indirect Window Object. Note that the size *does not refer* to the size of the popup window since the size of the popup window is determined in the Window Settings dialog box. This size refers to the area of the Indirect Window Object which is analogous to a box in which the popup window is to be placed. If the size is too small, then the popup window will appear clipped when displayed. Similarly, if the size of the Indirect Window Object is larger than the actual size of the popup window, the excess space will be blank.

Tips and Suggestions

Having trouble deciding which method to use to display a base window? Here are some suggestions:

- **If you want to clear the HMI display of all open windows and display a new full screen window**, then use the PLC Control Object function or select the Change Window option in the Function Key Object. The PLC Control Object is global (meaning it does not matter which windows are currently on display), so the HMI will always monitor the PLC address that you have selected. The Function Key Object can be made local to one or more windows by placing the object on only those windows. It can also be made global (meaning the HMI operator can always change to this window no matter which window is currently displayed), by placing the Function Key Object on the Common Window or the Fast Selection Window in the Task Bar, (more about these later in this chapter).
- **If you want to display a popup window on one particular full screen window**, then try using the Popup Window option in the Function Key Object. This is particularly useful if you want the HMI operator to control the ability to display the popup window. If you want the PLC to determine when to display the popup window, then try using the Direct Window object.

- **If you want to display a popup window that can be shown on any full screen window**, then try using the Popup Window option in the Function Key Object. You should place the Function Key Object in the Fast Selection Window with the Task Bar enabled, (more information is available on the Task Bar later in this chapter). This will allow the HMI operator access to that popup window regardless of which full screen window is displayed. If you want to display the popup window only when some condition in the PLC has occurred, then try using the Direct Window object on the Common Window, (more about the common window later in this
- chapter).

  If you want to display one of many possible popup windows on any full screen window, then try using the Indirect Window object on the Common Window. You can then let the PLC determine which popup window should be displayed or you can create several Set Word objects to allow the HMI operator to select which window to look at, (more about the Set Word
- object in a succeeding chapter).

  If you want to display a child popup window within a parent popup window, then it is best to use either the Indirect or Direct Window object not the Function Key object, even when you want the HMI operator to press a key to display the child popup window. Using the Function Key object causes the popup window to be displayed at the Window Setting position regardless of the location
  of the parent window. This means that the child window could appear outside of the parent window's area, especially if the parent window can be moved about the HMI screen. By using the Indirect or Direct Window object, the child window will always be located within the parent window even when the parent window has moved. To allow the HMI operator to press a

Creating a moving window

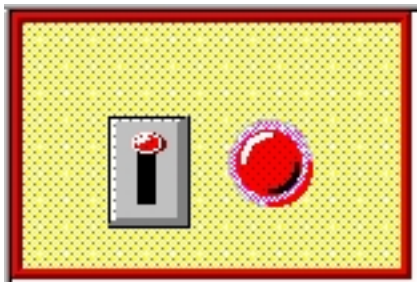
The popup windows that you create for display on the HMI can be configured so that the HMI operator can touch the window and touch somewhere else on the HMI display to move the window to that location. To create a popup window that can be moved, you must perform the following steps:

1. Create a popup window
2. Add a Function Key object with the Window Bar attribute in the popup window

The best way to show how this is done is to go through an example.

Creating a moving window

1. Create a new window that is 150 x 100 pixels similar to Window #13 shown below:



   Note that it really doesn't matter what type of objects are displayed in the window.
2. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box is displayed.
3. Click Window Bar.
4. Click the **Shape** tab. Click **Use Shape**. Click on the **Shape Library…** command. Select **Shape#0** from the **button1** library. Click **OK** to return to the Create Function Key Object dialog box.
5. Click the **Label** tab. Click the **Use label** checkbox.
6. Enter the text 'Window #13' into the **Contents** box. Click **OK** to exit the Create Function Key Object dialog box and go back to the EasyBuilder main screen.

7. Place the new Function Key object somewhere on Window #13. Reenter the Function Key Attributes dialog box by double-clicking on the Function Key object.
8. Click the **Profile** tab. Select **X=4** and **Y=4**. Select **Width=119** and **Height=21**.
9. Click **OK**.
10. Window #13 should now look something like this:

The Window Bar option in the Function Key Object allows the popup window to move about the HMI screen. When Window#13 is displayed, the HMI operator is able to move the popup window about the screen by touching the Window Bar on the top of Window#13 then touching another area of the HMI display where the Window should be relocated.

Creating a popup window that can be minimized

By using the Task Bar, popup windows you create for display on the HMI can minimized to a small icon that is displayed in the Task Bar. To create a popup window that can be minimized, you must perform the following steps:

- Create a popup window
- Add a Function Key object with the Window Bar attribute in the popup window
- Enable the Task Bar (**Edit Menu** =>**System Parameters**, select **General** tab in the **Task button** area select **Enable** from the **Attribute** pull down menu.)
- Add a Function Key object with the Minimize Window attribute in the popup window (optional)

Once more, an example is in order. Enabling and using the Task Bar are discussed later in this chapter.

▶ Creating a window that can be minimized

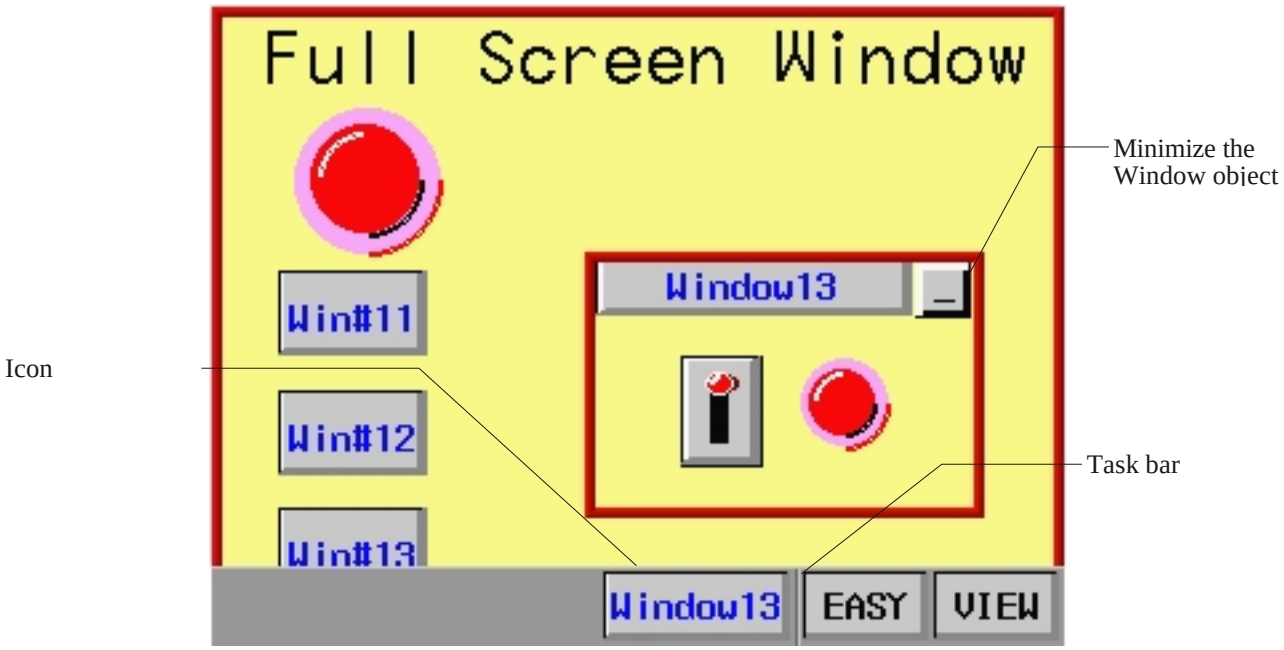
1. Use Window #13 from the last example:

2. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box is displayed.
3. Click Minimize Window.
4. Click the **Shape** tab. Click **Use Shape**. Click on the **Shape Library…** command. Select **Shape#0** from the **button1** library. Click **OK** to return to the Create Function Key Object dialog box.
5. Click the **Label** tab. Click the **Use label** checkbox.
6. Enter a dash mark '-' into the **Contents** box. Click **OK** to exit the Create Function Key Object dialog box and go back to the EasyBuilder main screen.

7. Place the new Function Key object somewhere on Window #13. Reenter the Function Key Attributes dialog box by double-clicking on the Function Key object.
8. Click the **Profile** tab. Select **X=124** and **Y=4**. Select **Width=21** and **Height=21**.
9. Click **OK**.
10. Window #13 should now look something like this:

The Window Bar option in the Function Key Object allows the popup window to be minimized into an icon. When the Task Bar is enabled, displaying Window#13 causes an icon to appear inside the Task Bar. The icon is labeled with the title that appears in the Window Bar of Window#13. By double-clicking the icon, the window will disappear from the HMI screen. The Minimize Window option in the Function Key Object accomplishes the same action when it is pressed.

To maximize the window, the HMI operator presses the icon. The popup window reappears in the last position that it was moved to.



Returning to a previous window

You can configure a function key to display the full-sized base window that was on the HMI screen before the currently shown window. Perform the following:

Create a function key to return to previous window

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box is displayed.
2. Click Return to Previous.
3. Configure the rest of the function key, and then click **OK**.

4. Place the function key object onto the full-sized base screen.

Placing a 'return to previous' function key onto a popup window will not work.

Using a function key to close a window

You can configure a function key to close any popup window that is currently displayed on the HMI screen:

Create a function key to close a window

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box is displayed.
2. Click Close Window.
3. Configure the rest of the function key, and then click **OK**.
4. Place the function key object onto the popup window.

Placing a 'close window' function key onto a full-size base screen will not work.

Using the Common Window

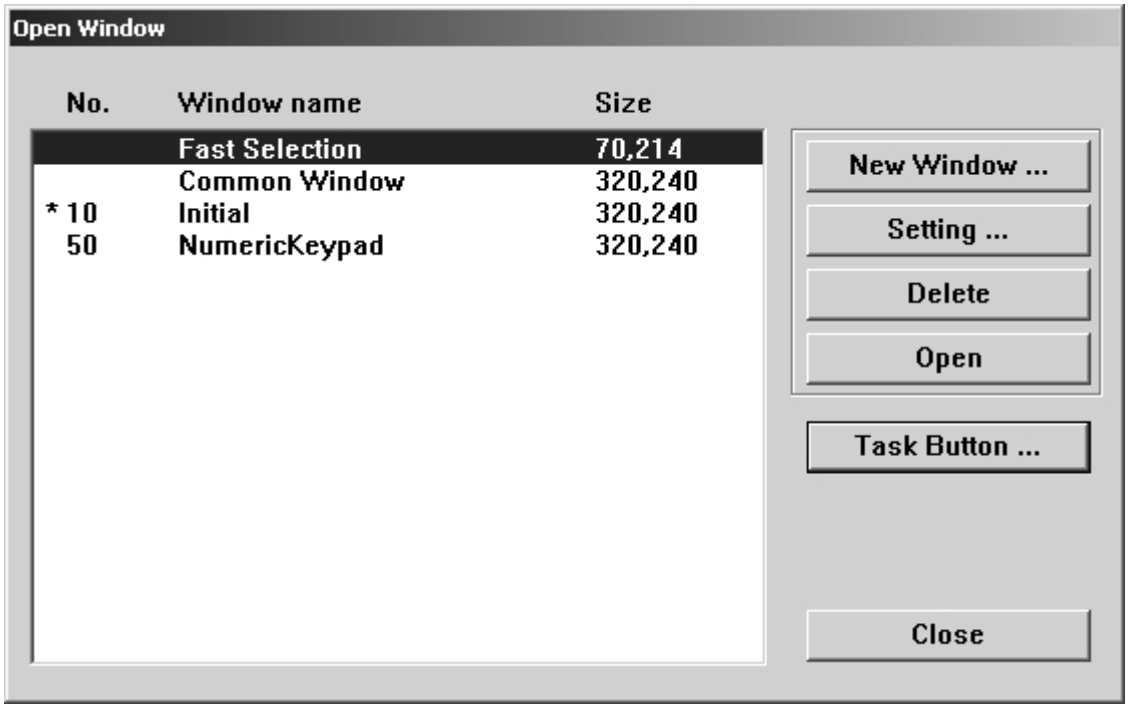
Your project might require that some data be displayed on the HMI screen at all times, regardless of which window(s) are displayed. For example, you may want to display a company logo on the HMI screen at all times, or you may want to display some critical data or an alarm message which should be seen no matter what windows are displayed.

Using base windows to display this information requires that you configure every full screen base window with the same graphics object. The common window however, is a predefined window in the HMI that you can enable to display this information. When created, the common window always operates in the background as a full screen window that overlays any full screen base window displayed.

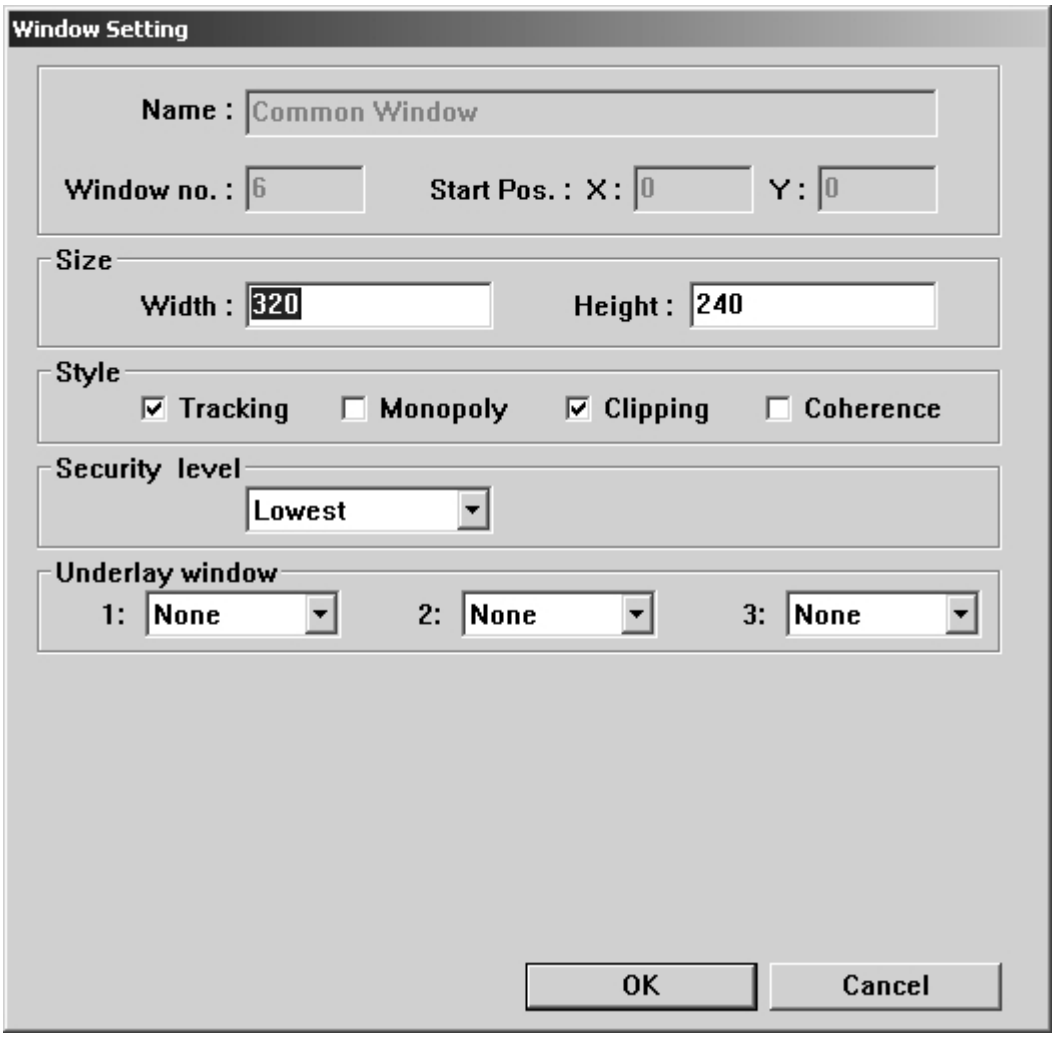
Whenever you create a new project, the common window (Window #6) is automatically created.

To Access the common window

1. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.



2. Highlight the **Common Window** and click **Settings**. The Window Setting dialog box appears.



3. As you can see, most of the parameters for a common window are disabled and cannot be changed. Window #6 is always reserved for the common Window. Although the **Style** settings are active, none of these settings affects the operation of the common window.
4. Press **OK**. The Open Window dialog box reappears with the Common Window selected.
5. If you wish to open the common window, click **Open**. Otherwise, click **Close** to return to the main screen of EasyBuilder.

Optional settings

Displaying popup windows from the common window

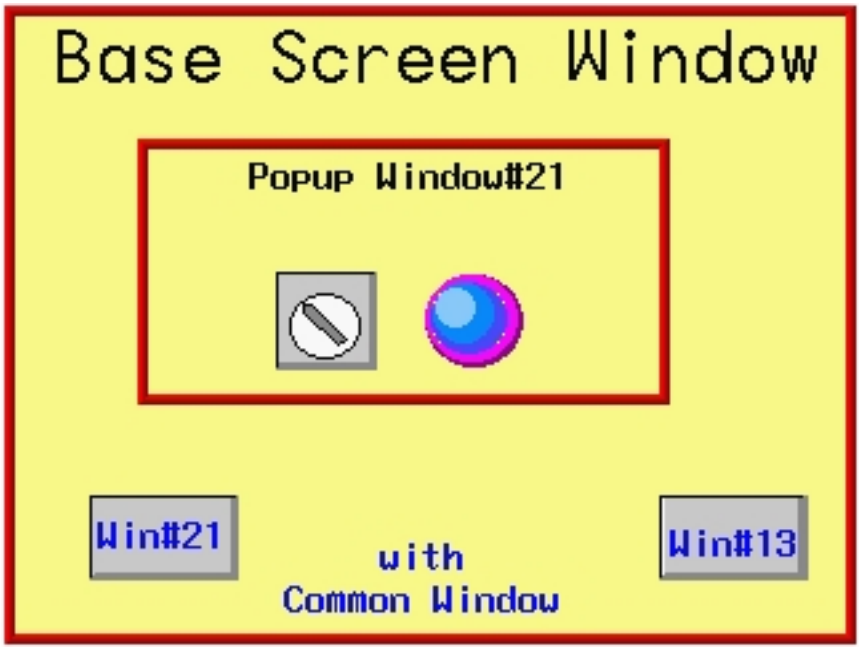
As with full screen base windows, popup windows can be called from the common window. In the **System Parameters** settings, you have the option of selecting any popup windows called from the common window to be displayed normally or to display above any popup windows called from a base window. If the **Normal** setting is selected, then the popup window called from the common window is displayed on top of any popup windows already on the HMI display. Popup windows displayed after common popup window are displayed over it.

If the **Above any others** setting is selected, then the popup window called from the common window is displayed above any popup windows that may be called from a base window.

To set the popup window option for a common window

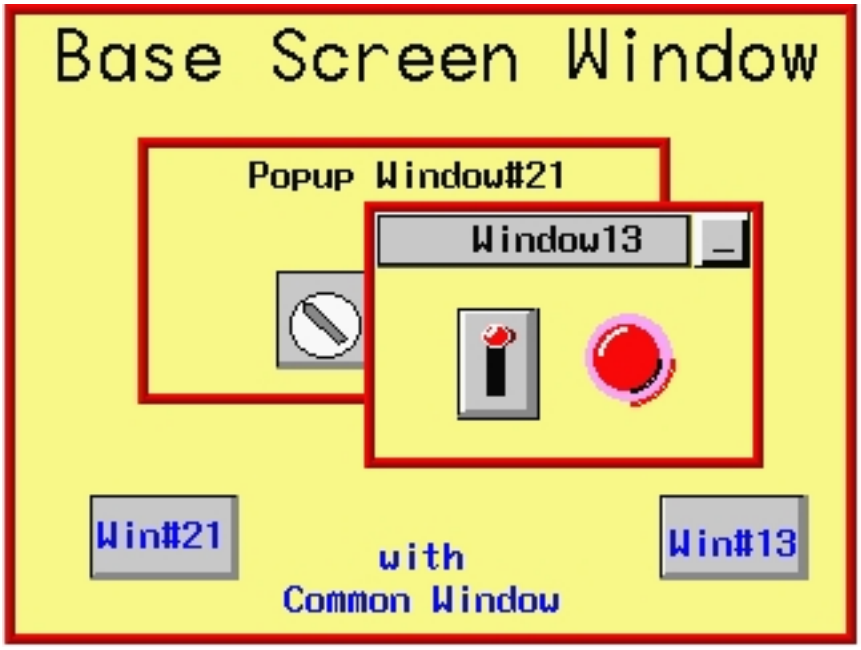
1. From the **Edit** menu, select **System Parameters**. The Set System Parameters dialog box appears.
2. Select the **General** tab. In the **Common Window** frame box, select either **Normal** or **Above any others** for the **Popup Window** box.
3. Click **OK** to return to the main screen of EasyBuilder.

The following illustration shows how this setting affects the operation of popup windows called from the common window.



Popup window #21 was called from the common window with the **Popup Window: Normal** setting.

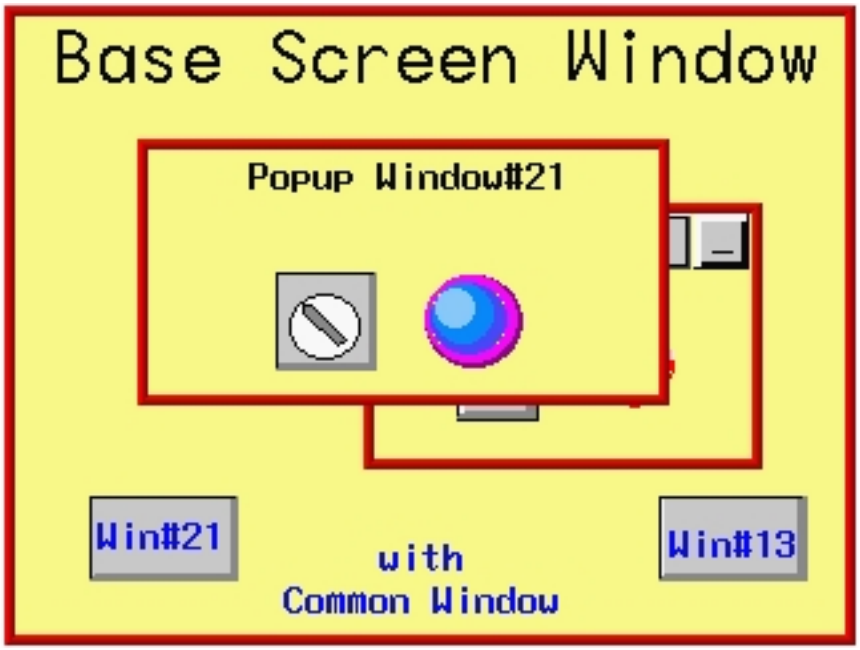
Window #13 is now called from the base window.



As expected, window #13 is displayed over window #21. The entire area of window #21 can be seen if the HMI operator activates it by pressing it.

Now suppose that the same scenario is played out with the **Popup Window: Above any others** setting.

Window #13 is now called from the base window, as before.



Notice that although window #13 was called *after* window #21, it is window #21 that remains above. Even when window #13 is activated by pressing it, window #21 continues to be the dominant window- always displaying above window #13.

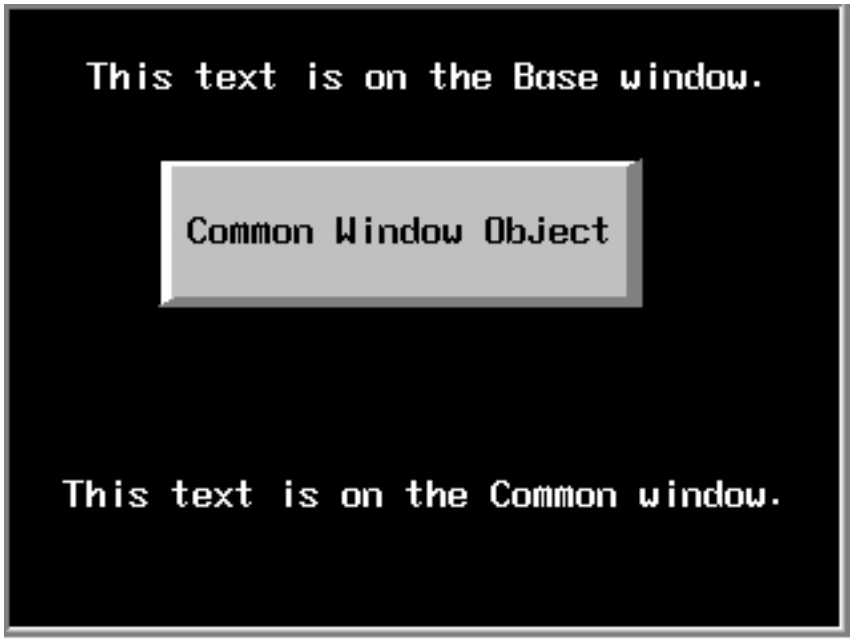
Displaying the common window above/below the base screen

This setting resolves the conflict that can occur if a graphics object on the common window occupies the same space on the HMI display as a graphics object on a base screen. Using the **Above base screen** attribute forces the graphics object on the common window to cover the base window graphics object. Using **Below base screen** has the opposite effect.

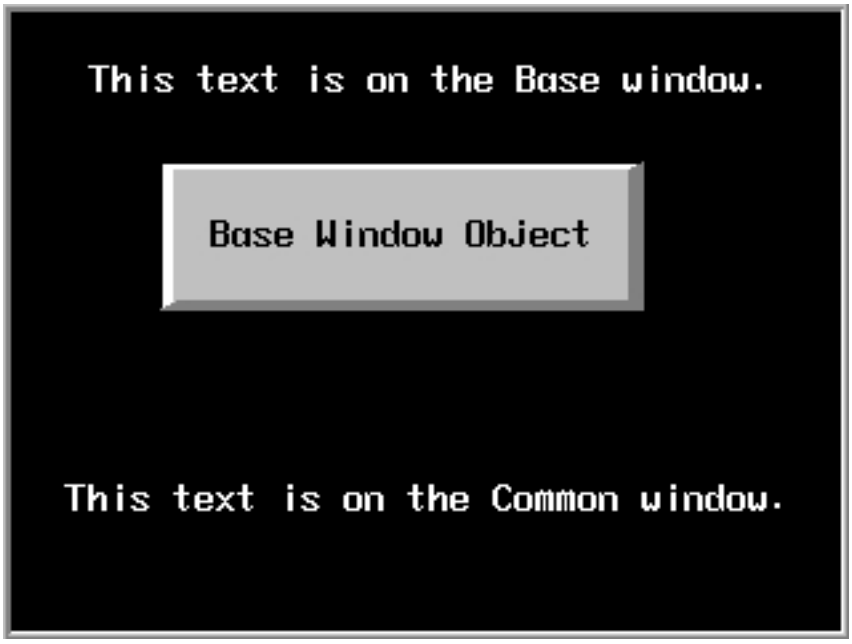
Common Window

Base Window

When **above base screen** is set, the result is:



When **below base screen** is set, the result is:



To set the above/below base screen option for a common window

1. From the **Edit** menu, select **System Parameters**. The Set System Parameters dialog box appears.
2. Select the **General** tab. In the **Common Window** frame box, select either **Above base screen** or **Below base screen** for the **Attribute** box.
3. Click **OK** to return to the main screen of EasyBuilder.

*Active graphics objects (objects that display information or graphics according to a data value in a PLC register or coil) take precedence over passive graphics objects (objects such as circles, lines, rectangles, etc.). Therefore, a Set Word object on a base window will cover a Rectangle object on a common window which occupies the same space even with the **Above base screen** attribute enabled.*

Changing the Active Common Window

Although only one common window can be on the HMI display at one time, you do have the ability to create multiple common windows by using base windows as common windows. Then, with the help of a Function Key object, the HMI operator can change the active common window.

The ability to change the common window adds more flexibility to your project should you need it. For example, you may have a series of full screen windows which all need a keypad for entry. Another series of full screen windows may require a common alarm message. By changing the common window with each series of windows, you can customize each common window to contain only the graphics objects that are needed.

To change the active common window

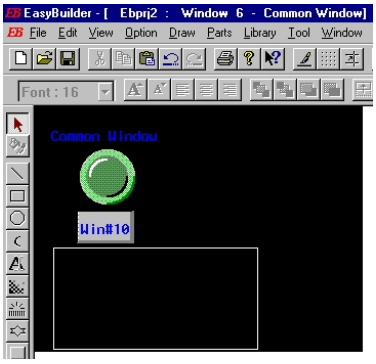
1. Create a common window.
2. Create a base window that is full screen. This will be used as another common window. Note that when a base window is used as a common window, the **Start Pos:**, **Style**, **Frame**,

and **Background** settings are ignored by the HMI. For this example, let's use Window#30 as the alternate common window.

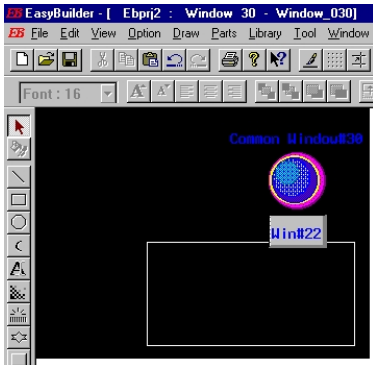
3. Create another full screen base window that can be displayed along with the common windows, (such as a startup window). Let's use Window#10, the initial window, for our example.
4. On Window#10, create a Function Key object that is used to change the common window. From the **Parts** menu, click **Function Key**. The Create Function Key Object dialog box appears.
5. On the General tab, click Change Common Window.
6. In the same frame, enter **30** for **Window No.**.
7. Click the **Shape** tab. Click **Use Shape**. Click on the **Shape Library…** command. Select **Shape#0** from the **button1** library. Click **OK** to return to the Create Function Key Object dialog box.
8. Click the **Label** tab. Click the **Use label** checkbox.
9. For our example, type **CCW** (meaning **C**hange **C**ommon **W**indow) into the **Contents** box. Click **OK** to exit the Create Function Key Object dialog box and go back to the EasyBuilder main screen.
10. Place the new Function Key object somewhere on Window #10.

Below is an illustration of what the default common window, Window#30 (alternate common window), and Window#10 (startup window) might look like:

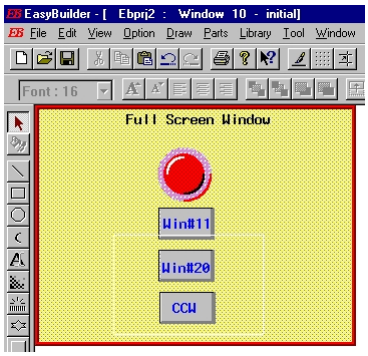
Default Common



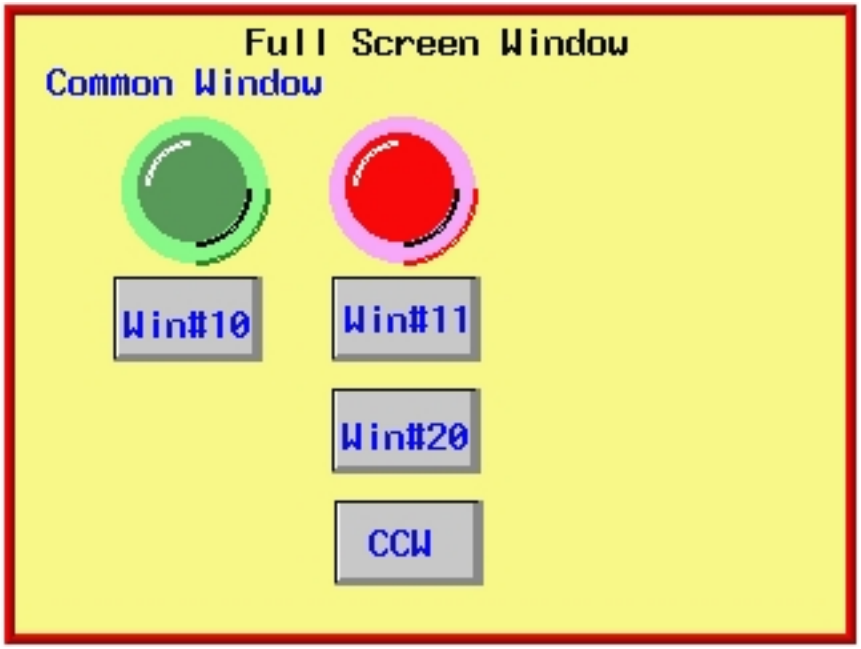
Window #30



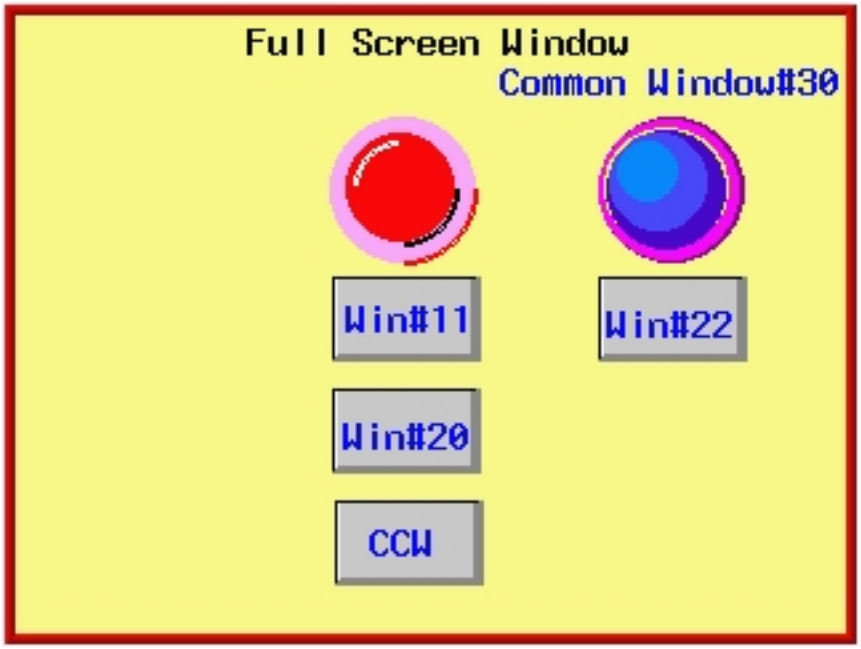
Window #10



When the HMI initializes, it displays the startup screen (Window #10) and the default common window.



To change common windows, we press the **CCW** function key that we have configured to change the common window to Window#30.



Window#30 remains the common window until another function key object is pressed that changes the common window or until the HMI is reset.

Using the Fast Selection Window

In the last section, you read that common windows are great for displaying information that should be displayed all the time, regardless of which base windows are active. However, you may create projects that require a window that is always accessible (like a common window) *but not always displayed*. For example, you may want a numeric

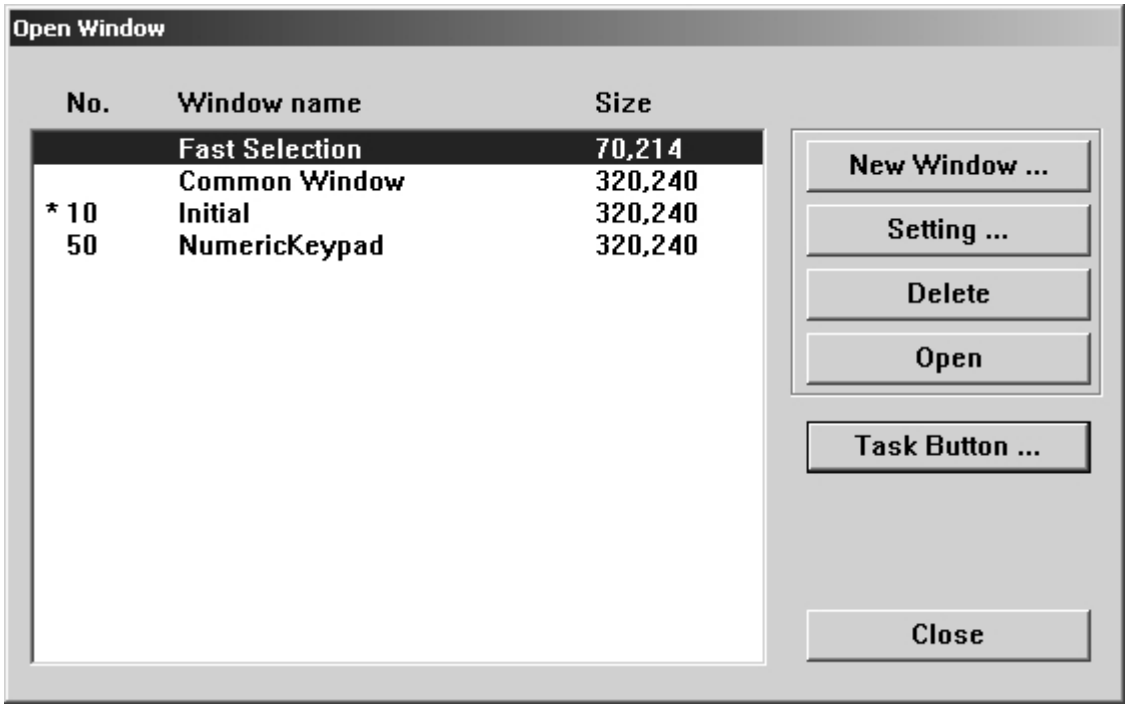
keypad available for any data entry. However, a keypad takes precious space on the HMI display. Ideally it should appear on screen only when a key was pressed and then, by pressing another (or the same) key, the keypad should disappear.

This is essentially the purpose of the Fast Selection window. The Fast Selection window is available only if the Task Bar is activated (more about this feature later in this chapter). The Fast Selection window can also be used as a menu key that allows the HMI operator to rapidly switch screens.

Whenever a new project is created, the fast selection window (Window #4) is automatically created.

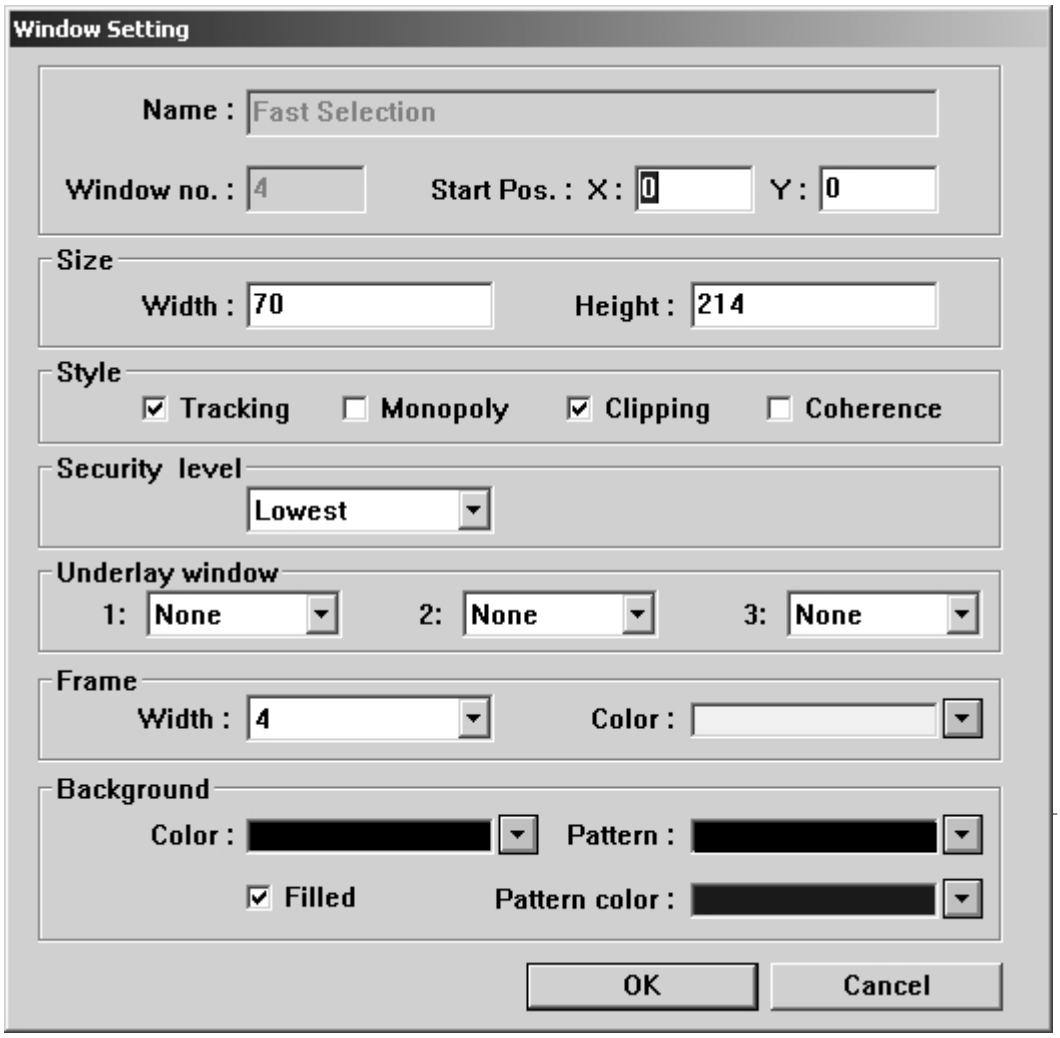
To access a fast selection window:

1. From the **Window** menu, select **Open Window**. The Open Window dialog box appears.



2. Highlight the Fast Selection Window and click Settings. The Window Setting dialog box appears.

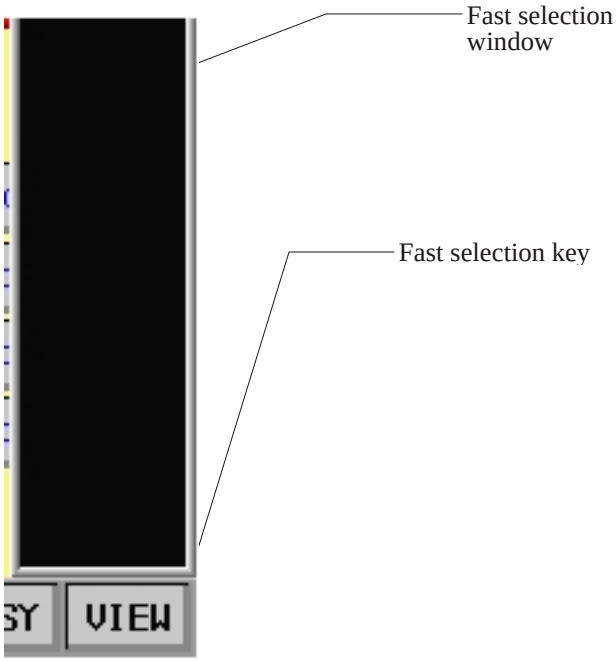
3. Click **Fast Selection**. The Window Setting dialog box appears.



3. The **Name:** and **Window No.:** are reserved to identify the Fast Selection window. The other parameters can be modified, (see the Window Settings section earlier in this chapter for more information). Notice that the default size is Width=70 and Height=214. The default was selected to create a 'sidebar' that contains function keys to display other windows. The example below illustrates how the Fast Selection Window can be used for this purpose. 4. Press **OK**. The Open Window dialog box reappears with the Fast Selection window selected.
5. If you wish to open the fast selection window, click **Open**. Otherwise, click **Close** to return to the main screen of EasyBuilder.

Using the Fast Selection Key

The fast selection key is used to display the Fast Selection window. Pressing the key again causes the Fast Selection window to close.



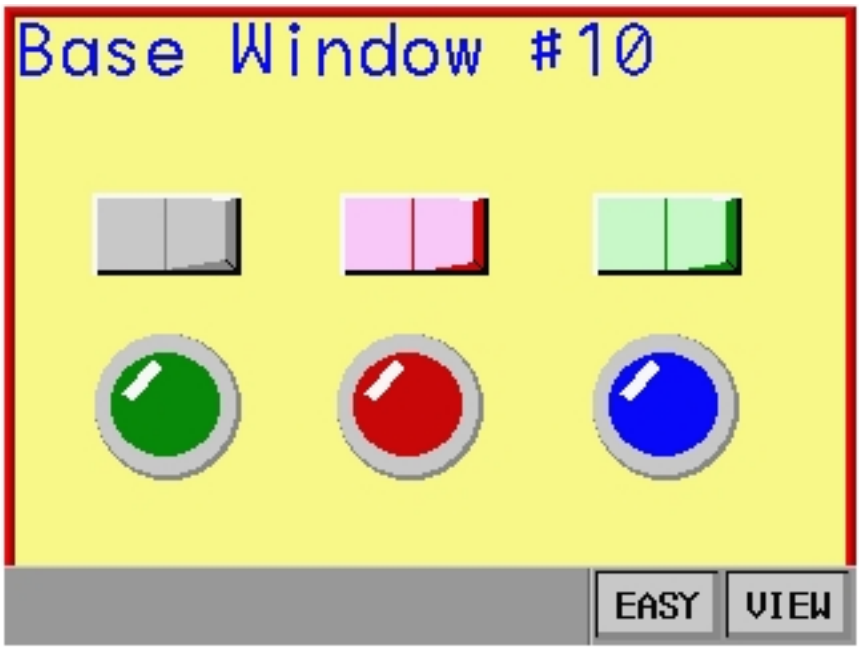
Changing screens using the Fast Selection window

One of the more useful purposes of the Fast Selection window is to use it as a 'menu' key for switching screens on the HMI display. The following example illustrates how you might create a Fast Selection window for this purpose. The example steps you through creating six full-screen base windows, enabling the Task Bar, and creating a Fast Selection window to call up the six screens.

We will start with the following six screens.

Next, create a new Fast Selection window using default settings. For this example, we will create six Function Key objects that change windows. Configure each **Function Key** object with the **Change Window** attribute and place the object into the Fast Selection window.

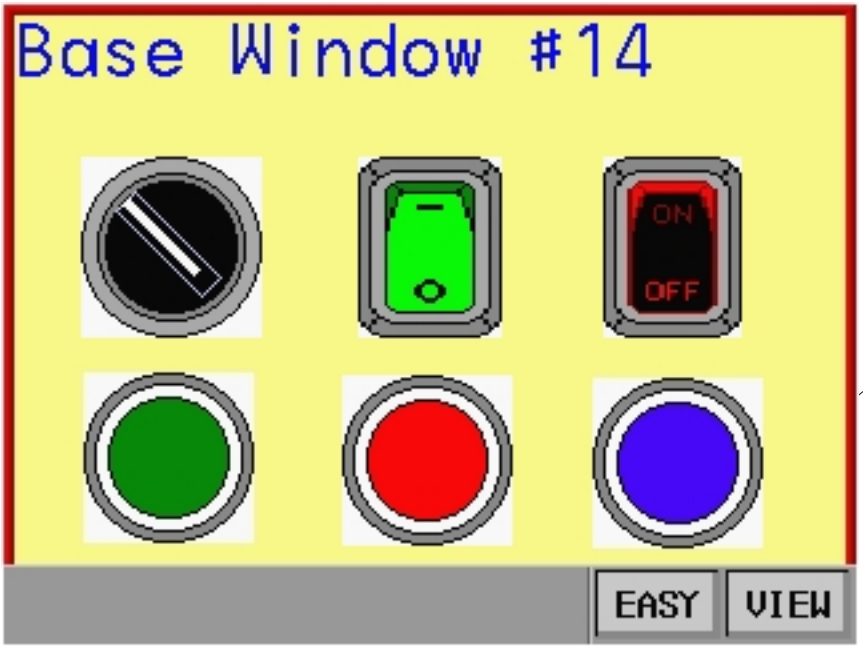
Finally, the Task Bar must be enabled, (see the next section for instructions). After downloading the project file to the HMI, the following initial screen should be shown.



Press the Fast Selection key on the Task Bar to activate the Fast Selection window.

Press one of the Function Keys to switch to another full screen window.

Finally, pressing the Fast Selection Key will remove the Fast Selection Window until it is needed again.



As you can see, the Fast Selection window, though not seen, is always readily available.

The Fast Selection window can only call full screen windows not popup windows. Therefore, you cannot use Function Key objects with the Popup Window attribute in Fast Selection windows.

Changing the Fast Selection Window

As with common windows, only one fast selection window can be on the HMI display at one time. However, you can create multiple fast selection windows by using base windows. Then, with the help of a Function Key object, the HMI operator can change the active fast selection window.

This feature allows you to customize each fast selection window created to perform a specialized function depending upon which set of windows is currently displayed. For example, for a group of data entry screens, you may want the fast selection window to contain a numeric keypad. For a group of data monitoring screens, you may want the fast selection window to contain a trend graph.

To change the active fast selection window

1. Create a fast selection window. **Note:** To use this feature, you must change the default width setting of the fast selection window from 70 to at least a minimum of 100. This is because the minimum allowable width size for base windows is 100.
2. Create a base window that is the same width and height as the fast selection window you just created. This will be used as another fast selection window. Match all settings to that used on the fast selection window. For our example, we will use Window#11.
3. Create a full screen base window that can be used to change the fast selection windows, (such as a startup window). Let's use Window#10, the initial window, for our example.
4. On Window#10, create a Function Key object that is used to change the fast selection window. From the **Parts** menu, click **Function Key**. The Create Function Key Object dialog box appears.
5. On the **General** tab, click **JOG FS-Window**.
6. In the same frame, enter **11** for **Window No.**.

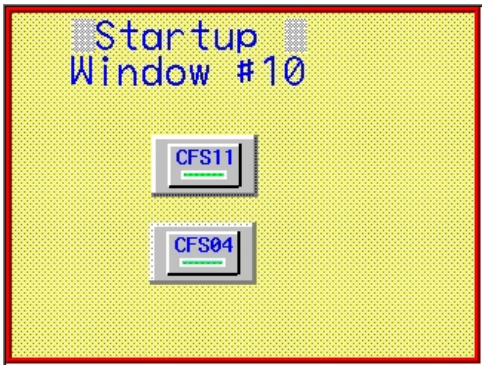
7. Click the **Shape** tab. Click **Use Shape**. Click on the **Shape Library…** command. Select **Shape#0** from the **button1** library. Click **OK** to return to the **Create Function Key Object** dialog box.
8. Click the **Label** tab. Click the **Use label** checkbox.
9. For our example, type **CFS11** (meaning **C**hange **F**ast **S**election Window#**11**) into the **Contents** box. Click **OK** to exit the Create Function Key Object dialog box and go back to the EasyBuilder main screen.
10. Place the new Function Key object somewhere on Window #10.
11. Now create another Function Key object with the JOG-FS attribute but select **4** for **Window number**
12. For the label, enter **CFS04** (meaning **C**hange **F**ast **S**election Window#**04**), which is the default Fast Selection window.
13. In the **Edit-System Parameters** menu under the **General** tab section, enable the task bar.
14. Save, compile, and download the project to your HMI.

Below is an illustration of what the default fast selection window, Window#11 (alternate fast selection window), and Window#10 (startup window) might look like:

Default Fast Selection Window

Alternate Fast Selection Window#11

Startup Window #10



When the HMI initializes, it displays the startup screen (Window #10) and the task bar.

Press the Fast Selection key on the task bar to view the default Fast Selection window.

To change fast selection windows, press the **CFS11** function key that has been configured to change the fast selection window to Window#11.

Window #11 will remain the Fast Selection window until a function key with the Jog-FS attribute is used to change the Fast Selection window again or the HMI resets. To change back to the default fast selection windows, we press the **CFS04** function key that we have configured to change the fast selection window back to the default fast selection window (Window #4).

Using the Task Bar

The Task Bar has three purposes:

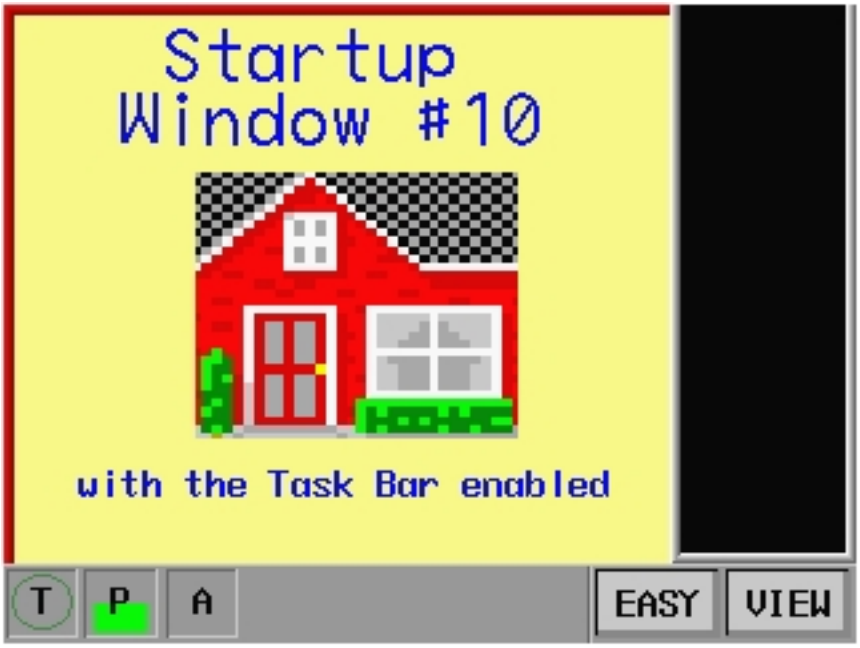
- Allows you to minimize popup windows to an icon that is placed in the Window bar.
- Allows you to access the Fast Selection window that you create.
- Displays information about the activity of your HMI, as an option.

The Task Bar is optional and must be enabled to see it on the HMI screen.

To activate the task bar

1. From the **Edit** menu, select **System Parameters**. The Set System Parameters dialog box appears.
2. Click the **General** tab to display the General form.
3. In the **Task Button** frame, click **Enable** in the **Attribute** box.

When enabled, the Task Bar should look like the following illustration:



Using the Window Bar

The window bar is where the minimized popup window icons reside. You can display a maximum of six popup windows at one time and every time a popup window is displayed on the HMI screen, an icon representing that window is displayed on the window bar of the task bar.

The window bar can be displayed or removed by pressing the Window Bar Key labeled **Easy**. Besides containing the popup window icons, the Window Bar also contains three optional indicators: the **Touch**, **CPU**, and **Alarm** indicators

The Touch Indicator 

This optional feature provides you with additional feedback when pressing the touch screen of the HMI. One of three colors is displayed in the circle, depending upon what area of the touch screen is touched. The three areas are:

- **Non-configured area** – Any area on the touch screen that does not contain an active touch screen object.
- **Active area –** Any area that contains an active touch screen object which, when pressed, executes some action.
- **Inactive area** – Any area that contains an active touch screen object but the active object is not able to execute its intended function. For example, a Function Key with the Popup Window option is defined as an active area when first pressed since it causes the popup window to be displayed. However, once the popup window is displayed, pressing this key serves no purpose and then it becomes an inactive area.

To activate the touch indicator

1. From the **Edit** menu, select **System Parameters**. The **System Parameter Setting** dialog box appears.
2. Click the **General** tab to display the General form.
3. In the **Task Button** frame, click **Enable** in the **Attribute** box.
4. Click the **Indicator** tab to display the Indicator form.
5. In the **Touch Indicator** frame, click **Enable** in the **Attribute** box.
6. Select which colors you would like to use for the **Non-Configured Area**, the **Inactive Area**, the **Active Area**, and the **Frame** box.
7. Click **OK** to exit **the System Parameters** dialog box.

The CPU Indicator 

This optional feature reflects the status of the HMI's microprocessor. As the color indicator grows, the microprocessor of the HMI is processing more data. The CPU indicator could be used as a rough indicator of how much activity is occurring in the HMI.

To activate the CPU indicator

1. From the **Edit** menu, select **System Parameters**. The Set System Parameters dialog box appears.
2. Click the **General** tab to display the General form.
3. In the **Task Button** frame, click **Enable** in the **Attribute** box.
4. Click the **Indicator** tab to display the Indicator form.
5. In the **CPU Indicator** frame, click **Enable** in the **Attribute** box.
6. Select the color you would like to use.
7. Click **OK** to exit the **System Parameters** dialog box.

The Alarm Indicator 

This optional feature reflects the status of alarms. The color indicator grows every time a new alarm condition becomes active. The color indicator is full when ten alarms are active. The Alarm indicator can be used as a rough indication of how many alarms are active. This can be used to notify the HMI operator when an alarm has occurred so that he may go to some alarm screen to get more information.

To activate the Alarm indicator

1. From the **Edit** menu, select **System Parameters**. The Set System Parameters dialog box appears.
2. Click the **General** tab to display the General form.
3. In the **Task Button** frame, click **Enable** in the **Attribute** box.
4. Click the **Indicator** tab to display the Indicator form.
5. In the **Alarm Indicator** frame, click **Enable** in the **Attribute** box.
6. Select the color you would like to use.
7. Click **OK** to exit the **System Parameters** dialog box.

General Settings

In EasyBuilder, you can configure how the task bar looks when it is activated for a more customized look. From the **Window** menu, select **Open Window**. In the Open Window dialog box, click the **Task button…** dialog box. The **Window Setting** box appears:

The only parameters that apply to the Task Bar are the **Width** and **Height** settings. The default setting for the **Width** parameter is **100** but you can select from a range of 10-200. The default setting for the **Height** parameter is **30** but you can select from a range of 10-100.

Click on the **Window…** button to change settings for the Window Bar key. The Fast Selection key can be modified by clicking the **Screen…** button. The Window Attribute dialog box appears. You can change the shape used for the two hot keys or the label that identifies them. You can also change the speed at which these keys display the Window Bar and the Fast Selection Window.

Creating a Message Board

You may have an application for an HMI in which the HMI operator must be able to leave messages for someone else to read. The message board feature creates a popup window that can be displayed on the HMI for such a purpose. Once displayed, the HMI operator 'draws' on the touch screen to write messages. The messages remain on the message board until a touch screen command is issued to clear the message board or the HMI is reset. One message board may be created per project and it must be created from a popup window. You can only call up the message board using the Direct Window Object. The message board popup window can contain any graphics
objects and the HMI operator can write over any graphics objects that appear on the window.

1. Create a Base Window that is partially sized, (to be used as a popup window).
2. From the **Edit** menu, click **System Parameters**. The Set System Parameters dialog box is displayed.
3. Click the **General** tab to display the General form.
4. Enter the number of the base window just created in **Message board window (0, 10-1999):** which is located at the bottom of the General form. Note: 0 is used to disable the message board feature.
5. Click **OK**.
6. Use the Function Key Object, Direct Window Object, or Indirect Window Object to call up the message board.

Optional Parameters

In addition to using the message board, the HMI operator can change the following parameters.

Set the Operation Mode for the Message Board

Using the Function Key Object, you can program a touch screen object to change 'operating mode' of the pen. There are three operating modes:

- Pen: select this mode when you want to write a message.
- Brush: select this mode when you want to erase parts of a message.
- Clip: select this mode to 'clip' or cut out a rectangular section of a message.

The touch screen object can be placed onto the message board or any other window.

To create a Change Message Board Operation Mode key

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box appears.
2. Click the **General** tab to display the General form.
3. Click the **Message Board** option.
4. Select **Set operation mode** from the pull-down.
5. Click the **Attributes** button to select the operation mode you want to use.
6. Click **OK** to exit the **Create Function Key Object** dialog box.
7. Place the new touch screen object on the window screen.

When this key is pressed, it will change the operation mode used when a message is written on the board.

Set the Pen Style for the Message Board

Using the Function Key Object, you can program a touch screen object to change the thickness of the line used for the pen. The touch screen object can be placed onto the message board or any other window.

To create a Change Message Board Pen Style key

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box appears.
2. Click the **General** tab to display the General form.
3. Click the **Message Board** option.
4. Click the **Set operation mode** pull-down box and select **Set pen style**.
5. Click the **Attributes** button to select the line width you want to use.
6. Click **OK** to exit the **Create Function Key Object** dialog box.
7. Place the new touch screen object on the window screen.

When this key is pressed, it will change the line thickness used when a message is written on the board.

Set the Pen Color for the Message Board

Using the Function Key Object, you can program a touch screen object to change the color used for the pen. The touch screen object can be placed onto the message board or any other window.

To create a Change Message Board Pen Color key

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box appears.
2. Click the **General** tab to display the General form.
3. Click the **Message Board** option.
4. Click the **Set operation mode** pull-down box and select **Set pen color**.
5. Click the **Attributes** button to select the color you want to use.
6. Click **OK** to exit the **Create Function Key Object** dialog box.
7. Place the new touch screen object on the window screen.

When this key is pressed, it will change the color used when a message is written on the board.

Clear the Message Board

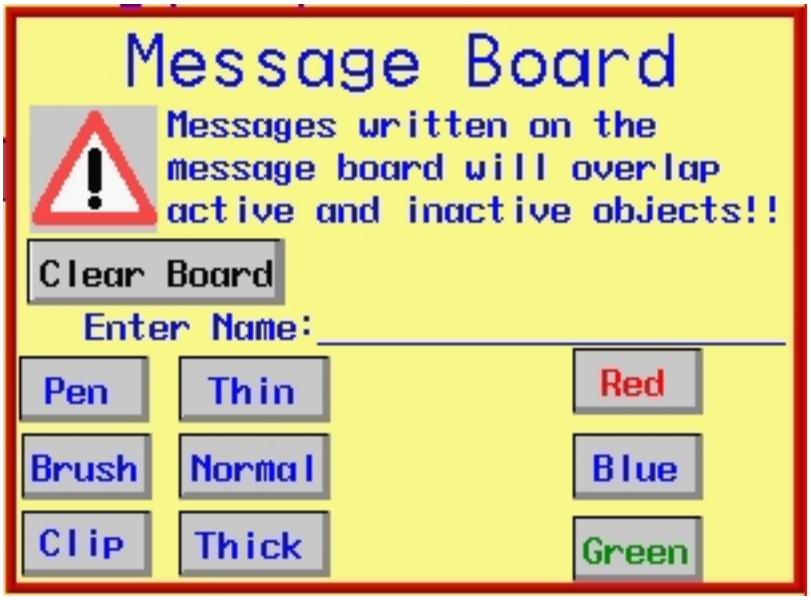
Using the Function Key Object, you can program a touch screen object to clear the message board. The touch screen object can be placed onto the message board or any other window.

To create a Clear Message Board key

1. From the **Parts** menu, select **Function Key**. The Create Function Key Object dialog box appears.
2. Click the **General** tab to display the General form.
3. Click the **Message Board** option.
4. Click the **Set operation mode** pull-down box and select **Clear board**.
5. Click the **Shape** tab to select the shape that you want to use.
6. Click the **Label** tab to select the label.
7. Click **OK** to exit the **Create Function Key Object** dialog box.
8. Place the new touch screen object on the window screen.

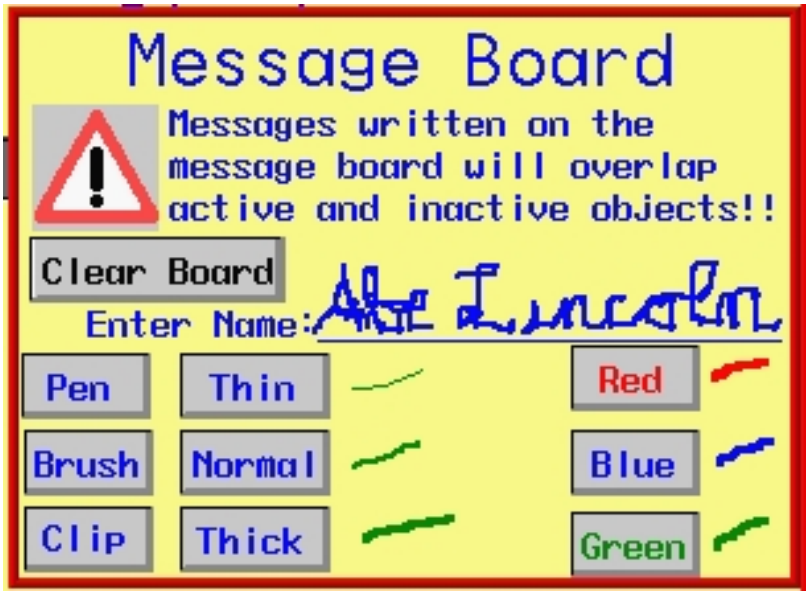
When this key is pressed, it will clear any messages written on the message board.

Below is an illustration of how a message board might look with all of the options available on the message board.



In this example, the HMI operator can change the operation mode of the message board by pressing the Pen, Brush, or Clip keys. The Pen Style can be changed by pressing the Thin, Normal, and Thick keys. The Pen Color is changed by pressing the Red, Blue, or Green keys. Finally, any messages written on the board by the HMI operator can be erased by pressing the Clear Board key.

To write on the message board, we recommend that you use some dull pointed device (such as a retractable pen) that won't mar or leave a permanent mark on the HMI's touch screen. Then practice your penmanship!



Chapter 6 - Creating Graphic Objects

Drawing Objects

To display any graphics objects on the HMI screen you must either create them using the drawing tools included with EasyBuilder, import them from another applications program, or select from the many choices available in the graphics libraries included with EasyBuilder.

This chapter shows you how to create, use, and save graphics objects in EasyBuilder. You have learned from previous chapters how graphics objects are placed onto windows. In this chapter, we concentrate on how to create graphics objects.

Using the Drawing Tools

Several drawing tools are provided in EasyBuilder to make it easier to create graphics objects. With these tools, you can create simple geometric shapes such as straight lines, circles, rectangles, and polygonal shapes. You can also combine several various geometric shapes to form complex shapes that can be stored into one of the graphics libraries for later retrieval.

The Line Tool

Use the Line tool to create straight lines on a window. Each line that you create has three parameters associated with it: length, thickness, and color.

To create a line 

1. From the **Draw** menu, click **Line** or click the **Line** icon in the Draw toolbar. The Attributes dialog box appears.
2. Click the pull down box from the Color box. The Color dialog box appears.
3. Click on the appropriate color box, then click **OK**. The color box should reflect what you have chosen.
4. Click on the line thickness that you want to use.
5. Move the mouse cursor over to the work area of EasyBuilder onto the window that you are currently editing. The mouse cursor has changed from an arrow to a crosshair cursor.
6. Click to mark the beginning of the line. Move the mouse to the location where the end of the line should be. Click the mouse cursor again to mark the end of the line. The line object is formed and displayed on the window screen with small white square blocks around the boundaries of the line.
7. To continue creating more lines repeat the last step. To move the line just created, click on the **Select** option from the **Edit** menu or click the **Select** icon in the **Draw** toolbar. Move the line to the preferred location.
8. To change any of the attributes, double-click on the line to display the **Line Object's Attribute** dialog box.

The Rectangle Tool

The rectangle tool is used to create rectangles or squares. Each rectangle created has four parameters associated with it: size, thickness, color, and filled.

To create a rectangle 

1. From the **Draw** menu, click **Rectangle** or click the **Rectangle** icon in the Draw toolbar. The Attributes dialog box appears.
2. If the interior of the rectangle is to be filled with a color, check the **Filled** box. Two additional settings appear.
3. If you want to fill the rectangle with a solid color, then select a color for the **Interior** box. Click **Pattern Style** to display the **Pattern Style** dialog box. Select the upper left-most

pattern for a solid color. If you want to use a pattern for the filled section of the rectangle, then select from the available patterns, and click **OK** to exit the **Pattern Style** dialog box. In the **Attributes** dialog box, select the **Pattern** color.

4. In the Frame section, click the pull-down arrow of the **Color** box and select a color. Click **OK** in the Color dialog box to go back to the Attributes dialog box.
5. Click on the line thickness that you want to use.
6. Move the mouse cursor over to the work area of EasyBuilder onto the window that you are currently editing. The mouse cursor has changed from an arrow to a crosshair cursor.
7. Click to mark a corner of the rectangle. Move the mouse to where the opposite corner of the rectangle should be. Click the mouse cursor again to mark the corner of the rectangle. The rectangle object is formed and displayed on the window screen with small white square blocks around the perimeter of the rectangle.
8. To continue creating more rectangles repeat the last step. To move the rectangle just created, click on the **Select** option from the **Edit** menu or click the **Select** icon in the **Draw** toolbar. Move the rectangle to the preferred location.
9. To change any of the attributes, double-click on the rectangle to display the **Rectangle Object's Attribute** dialog box.

The Ellipse/Circle Tool

The ellipse tool is used to create ellipses or circles. Each ellipse created has four parameters associated with it: size, thickness, color, and filled.

▶ **To create an ellipse** 

1. From the **Draw** menu, click **Ellipse/Circle** or click the **Ellipse/Circle** icon in the Draw toolbar. The Attributes dialog box appears.
2. If the interior of the ellipse is to be filled with a color, check the **Filled** box. Two additional settings appear.
3. If you want to fill the ellipse with a solid color, then select a color for the **Interior** box. Click **Pattern Style** to display the **Pattern Style** dialog box. Select the upper left-most pattern for a solid color. If you want to use a pattern for the filled section of the ellipse, then select from the available patterns, and click **OK** to exit the **Pattern Style** dialog box. In the **Attributes** dialog box, select the **Pattern** color.
4. In the Frame section, click the pull-down arrow of the **Color** box and select a color. Click **OK** in the Color dialog box to go back to the Attributes dialog box.
5. Click on the line thickness that you want to use.
6. Move the mouse cursor over to the work area of EasyBuilder onto the window that you are currently editing. The mouse cursor has changed from an arrow to a crosshair cursor.
7. Click to mark a corner of the ellipse. Move the mouse to where the opposite corner of the ellipse should be. Click the mouse cursor again to mark the ellipse. The ellipse object is formed and displayed on the window screen with small white square blocks around the perimeter of the ellipse.
8. To continue creating more ellipses repeat the last step. To move the ellipse just created, click on the **Select** option from the **Edit** menu or click the **Select** icon in the **Draw** toolbar. Move the ellipse to the preferred location.
9. To change any of the attributes, double-click on the ellipse to display the **Ellipse Object's Attribute** dialog box.

The Arc Tool

The arc tool is used to create arcs. Each arc created has three parameters associated with it: size, thickness, and color.

▶ **To create an arc** 

1. From the **Draw** menu, click **Arc** or click the **Arc** icon in the Draw toolbar. The Attributes dialog box appears.

2. In the Frame section, click the pull-down arrow of the **Color** box and select a color. Click **OK** in the Color dialog box to go back to the Attributes dialog box.
3. Click on the line thickness that you want to use.
4. Move the mouse cursor over to the work area of EasyBuilder onto the window that you are currently editing. The mouse cursor has changed from an arrow to a crosshair cursor.
5. Click to mark a corner of the arc. Move the mouse to where the opposite corner of the arc should be. Click the mouse cursor again to mark the arc. The arc object is formed and displayed on the window screen with small white square blocks around the perimeter of the arc.
6. To continue creating more arcs repeat the last step. To move the arc, click on the **Select** option from the **Edit** menu or click the **Select** icon in the **Draw** toolbar. Move the arc to the preferred location.
7. To change any of the attributes, double-click on the arc to display the **Arc Object's Attribute** dialog box.

The Polygon Tool

The polygon tool is used to create polygons. Each polygon created has four parameters associated with it: size, thickness, color, and filled.

To create a polygon 

1. From the **Draw** menu, click **Polygon**. Or click the **Polygon** icon in the Draw toolbar. The Attributes dialog box appears.
2. If the interior of the polygon is to be filled with a color, check the **Filled** box. Two additional settings appear.
3. If you want to fill the polygon with a solid color, then select a color for the **Interior** box. Click **Pattern Style** to display the **Pattern Style** dialog box. Select the upper left-most pattern for a solid color. If you want to use a pattern for the filled section of the polygon, then select from the available patterns, and click **OK** to exit the **Pattern Style** dialog box. In the **Attributes** dialog box, select the **Pattern** color.
4. In the Frame section, click the pull-down arrow of the **Color** box and select a color. Click **OK** in the Color dialog box to go back to the Attributes dialog box.
5. Click on the line thickness that you want to use.
6. Move the mouse cursor over to the work area of EasyBuilder onto the window that you are currently editing. The mouse cursor has changed from an arrow to a crosshair cursor.
7. Click to mark a corner of the polygon. Move the mouse to where another corner of the polygon should be. Click the mouse cursor again to mark the corner. Continue clicking each corner until it is time to connect the last corner to the first corner of the polygon. Right click to complete the polygon. The polygon object is formed and displayed on the window screen with small white square blocks around the perimeter of the polygon.
8. To continue creating more polygons repeat the last step. To move the polygon, click on the **Select** option from the **Edit** menu or click the **Select** icon in the **Draw** toolbar. Move the polygon to the preferred location.
9. To change any of the attributes, double-click on the polygon to display the **Polygon Object's Attribute** dialog box.

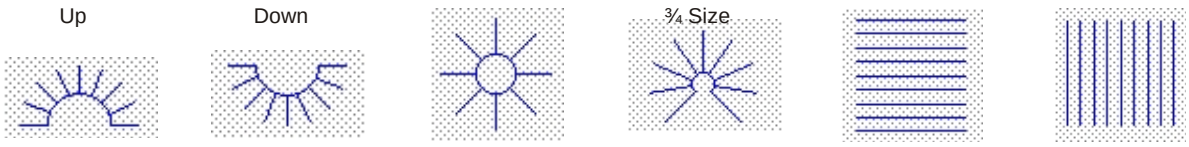
The Scale Lines Tool

The Scale Lines tool is used to create scales. Each scale created has four parameters associated with it: size, thickness, color, and filled.

To create a scale 

1. From the **Draw** menu, click **Scale**. Or click the **Scale** icon in the Draw toolbar
2. Move the mouse cursor over to the work area of EasyBuilder onto the window that you are currently editing. The mouse cursor has changed from an arrow to a crosshair cursor.

3. Click to mark a corner of the scale. Move the mouse to where the opposite corner of the scale should be. Click the mouse cursor again to mark the corner of the scale. The scale object is formed and displayed on the window screen with small white square blocks around the perimeter of the scale.
4. From the **Edit** menu, select **Change Attribute**. The Scale Object's Attribute dialog box appears.
5. In the Frame section of the Style tab, click the pull-down arrow of the **Color** box and select a color. Click **OK** in the Color dialog box to go back to the Scale Object's Attribute dialog box.
6. Click on the line thickness that you want to use.
7. In the Scale section, select from one of the six **Styles**:



8. Enter the number of divisions in the **Division:** box. For **Up, Down, Full,** and ¾ **Size** enter the **Meter Length**.
9. Adjust the size and position of the scale in the Profile tab, if necessary.
10. Click **OK**. The Scale Object's Attribute closes and the main screen of EasyBuilder reappears with the Scale Object displayed. Scale lines are most often used when creating bar graphs, scale meters, and trend graphs.

Using Text

EasyBuilder allows you to create text boxes using eight font sizes: 8, 16, 24, 32, 48, 64, 72, and 96. The text can be displayed using any of the 256 colors available in the MT506C, MT506H MT508C, or MT510H or the four shades available in the MT506M and MT510M. The text can be left, center, or right justified in the text box.

▶ To create a text box 

1. From the **Draw** menu, click **Text**. Or click the **Text** icon in the Draw toolbar. The Create Text Object dialog box appears.
2. Select which color you want in the **Color:** box.
3. Select the font size in the **Font:** box.
4. Select the type of justification in **Align:** box.
5. In the **Content:** box, enter the text that you would like to appear in the text box.
6. Click **OK**. The main screen of EasyBuilder reappears with the outline of the text box tagged to the cursor in the upper left corner of the work area.
7. Move the text box to the area you wish to place it on the window and click.
8. The text box appears with small square boxes around the perimeter. Click on another area of the window to end the edit session or double-click to reedit the text box parameters.

If you wish to create your own characters, you can use the EasyASCIIFontMaker program to edit the character sets. For more information, see the section "The EasyASCIIFontMaker" in Chapter 4 "Using EZware-500".

The following illustration shows the various font sizes available.

To change the font size

1. By clicking onto a text box, you can easily change the font size by using the **Font:** box or the **Enlarge** or **Reduce** buttons on the Manager toolbar.
2. Click on the pull-down box of the **Font:** box and select one of the font sizes or
3. Click the **Enlarge** button to increase the size of the text box by one font size.
4. Click the **Reduce** button to decrease the size of the text box by one font size.
5. Note that if the text box overlaps one of the borders of the window when enlarged, then the command to enlarge the text box is ignored.

Predefined Shapes and Bitmaps

In addition to using the drawing tools to create your own graphics, EasyBuilder provides two types of predefined graphics objects: shapes and bitmaps. These graphics objects are stored in libraries that come with EasyBuilder, (more on libraries in the next section). Shapes and bitmaps are both used for the same purpose, to display a
predefined complex graphics object on the HMI screen. The difference between shapes and bitmaps is how they are composed and stored.

Shapes are vector-based graphics which are stored in files that contain dimensional information about the shape. Shapes are actually composed of several simpler graphics like the ones created by the drawing tools. Because of this, they can usually be more easily modified than bitmaps.

Bitmaps are pixel-based graphics which are stored in files that contain information about each pixel to compose a bitmap graphic. The pixel is the smallest possible detail that you can change on a screen. Because of this, although

In most cases, you won't need to know the difference between a shape graphic and a bitmap graphic. You will base your selection on what looks good. However, keep the following comparisons in mind.

	Shapes	Bitmaps
Ease of Creation	Can be created using EasyBuilder	Must be created using a desktop graphics application such as Microsoft Paint™
Memory Requirements	Low	High
Quality of picture after changing size	Good	Fair to Bad
Time required to display on HMI screen	Quick if not too complex	Quick if not too large

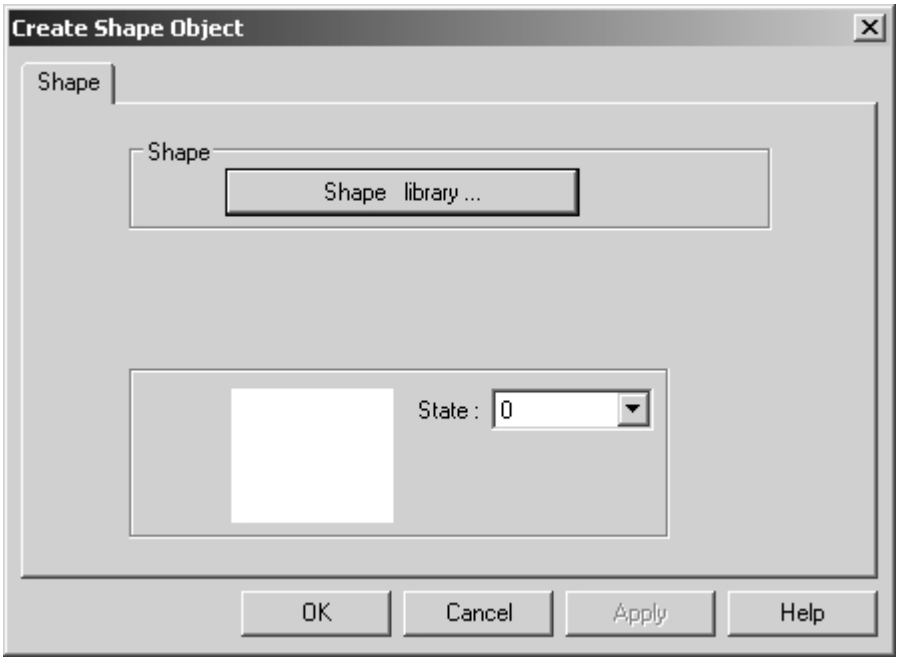
Shapes and bitmaps are used in both passive and active graphics objects. Each shape or bitmap can be resized or moved just like any drawing object. The following examples show how to place a pre-defined shape or a bitmap on the HMI screen as a passive object. Using shapes and bitmaps with active graphics objects is covered in Chapter 8 "Representing Data with Graphics Objects". Information on creating new shapes and bitmaps and storing them in libraries will be covered in a later section of this chapter.

Using a Predefined Shape

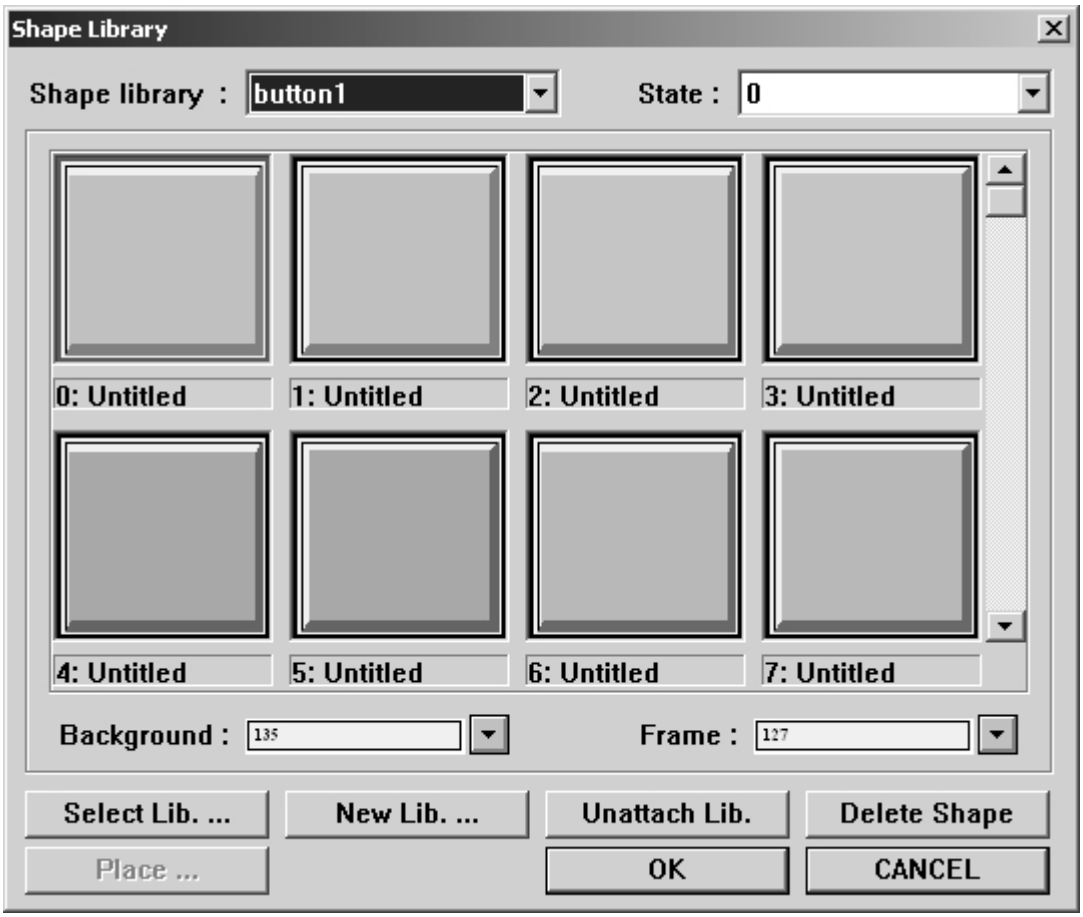
You can select from many shapes that are included in the EasyBuilder configuration software. Shapes can be used for active or passive objects. The following procedure describes using a passive shape object

To use a shape 

1. From the **Draw** menu, click **Shape**. Or click the **Shape** icon in the Draw toolbar. The Create Shape Object dialog box appears.



2. Click the **Shape library…** button. The Shape Library dialog box appears.



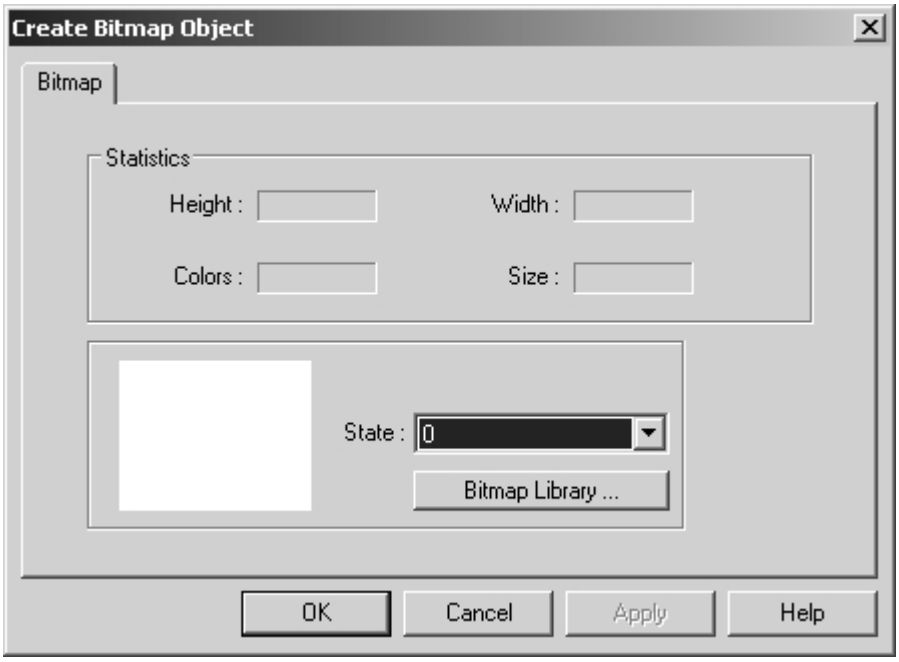
3. Click the pull-down box of **Shape library:** and choose from one of the open libraries of shapes.
4. Use the scroll bar to view available shapes. Click on the shape that you want.
5. Click **OK** The Create Shape Object dialog box reappears. The shape that you selected should be displayed in this dialog box.
6. Click **OK**. The main screen of EasyBuilder reappears with an outline of the shape object in the upper left corner of the work area.
7. Move the shape to the location desired and click. The shape appears with small square boxes around the perimeter of the shape.
8. The shape can be resized or you can edit the shape's parameters by double-clicking to enter the **Shape Object's Attribute** dialog box.

Using a Predefined Bitmap

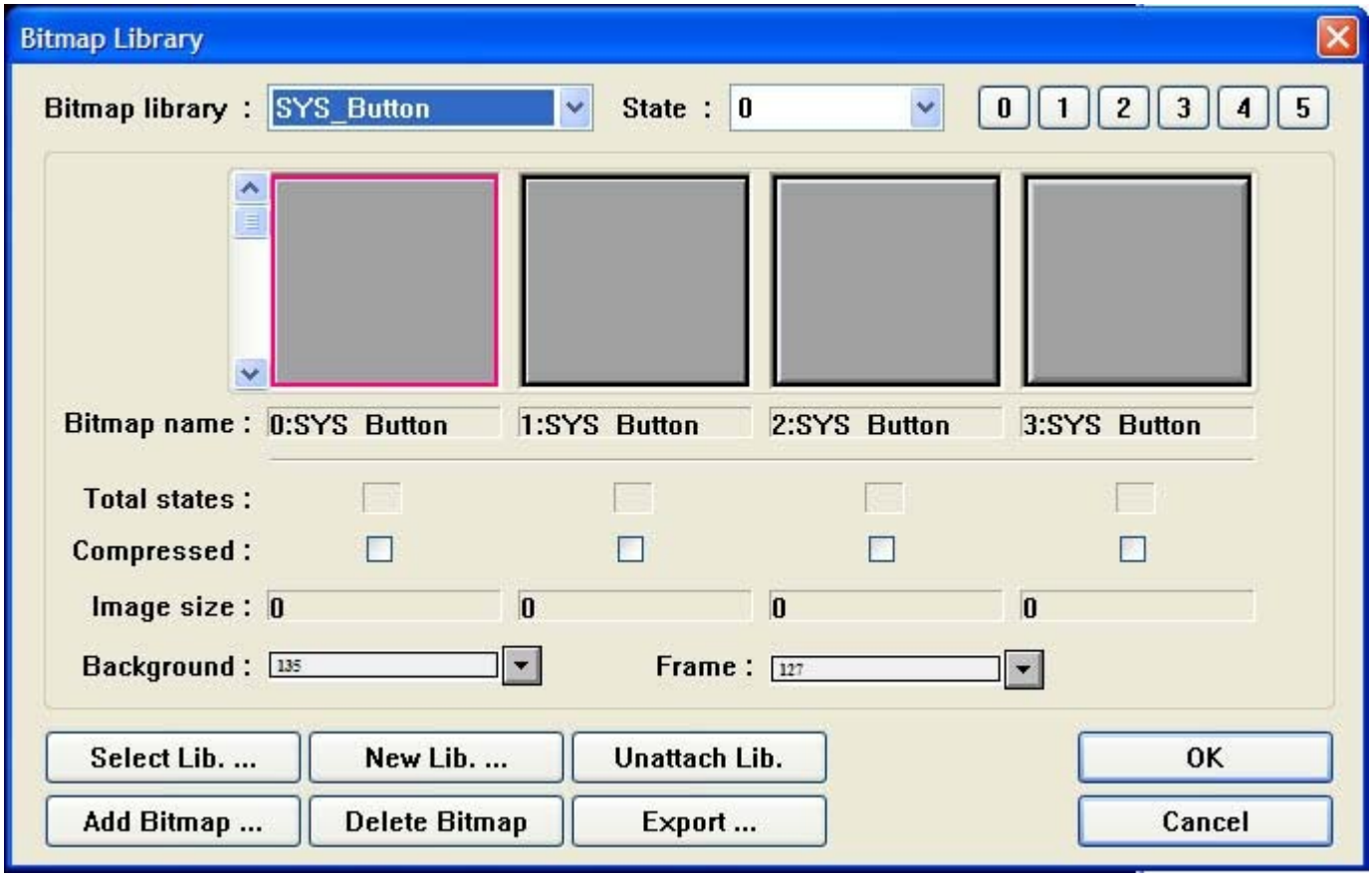
You can select from many bitmaps that are included in the EasyBuilder configuration software. Bitmaps can be used for active or passive objects. The following procedure describes using a passive bitmap object.

▶ To use a bitmap 

1. From the **Draw** menu, click **Bitmap**. Or click the **Bitmap** icon in the Draw toolbar. The Create Bitmap Object dialog box appears.



2. Click the **Bitmap library…** button. The Bitmap Library dialog box appears.



3. Click the pull-down box of **Bitmap library:** and choose from one of the open libraries of bitmaps.
4. Use the scroll bar to view available bitmaps. Click on the bitmap that you want.
5. Click **OK** The Create Bitmap Object dialog box reappears. The bitmap that you selected should be displayed in this dialog box.
6. Click **OK**. The main screen of EasyBuilder reappears with an outline of the bitmap object in the upper left corner of the work area.
7. Move the bitmap to the location desired and click. The bitmap appears with small square boxes around the perimeter of the bitmap.
8. The bitmap can be resized or you can edit the bitmap's parameters by double-clicking to enter the **Bitmap Object's Attribute** dialog box.

Graphics Libraries

Graphics Libraries are files that come with EasyBuilder which contain graphics shapes and bitmaps. Over 500 predefined shapes and bitmaps are included with EasyBuilder. In addition, you can create new bitmaps or shapes and add them to the libraries. You can even create new libraries as you need them. Each library contains a maximum of 48 shapes or bitmaps. Each shape or bitmap has a maximum of 32 'states' or pictures tied to it.

EasyBuilder allows a maximum of ten bitmap libraries and ten shape libraries to be open per project file. Finally, a group library can be used to store several graphics objects (such as a keypad) to be easily used when needed. For example, you may create a bar graph that could be used in several different projects. The bar graph can be stored into the group library for easy retrieval.

All libraries are accessed from the **Library** menu or by clicking the appropriate icon in the **Standard** toolbar.

What are 'states'?

Many of the graphics objects (ex. Set Word Object or Moving Shape Object) use states to determine which picture to display. Shapes and bitmaps can be configured with up to 32 different states.

Each shape can be configured to show a different picture depending upon a 'state'. States are used when the shape is tied to an active object that is monitoring a value in a PLC. For instance, you would use a two-state shape to represent a Bit Lamp object. You can also use a different picture of a shape for a passive object. Click the **State:** pull-down box in the Shape or Bitmap library and select a different number. Most of the predefined shapes in the libraries have at least two states.

Using Shape Libraries

EasyBuilder includes fourteen shape libraries:

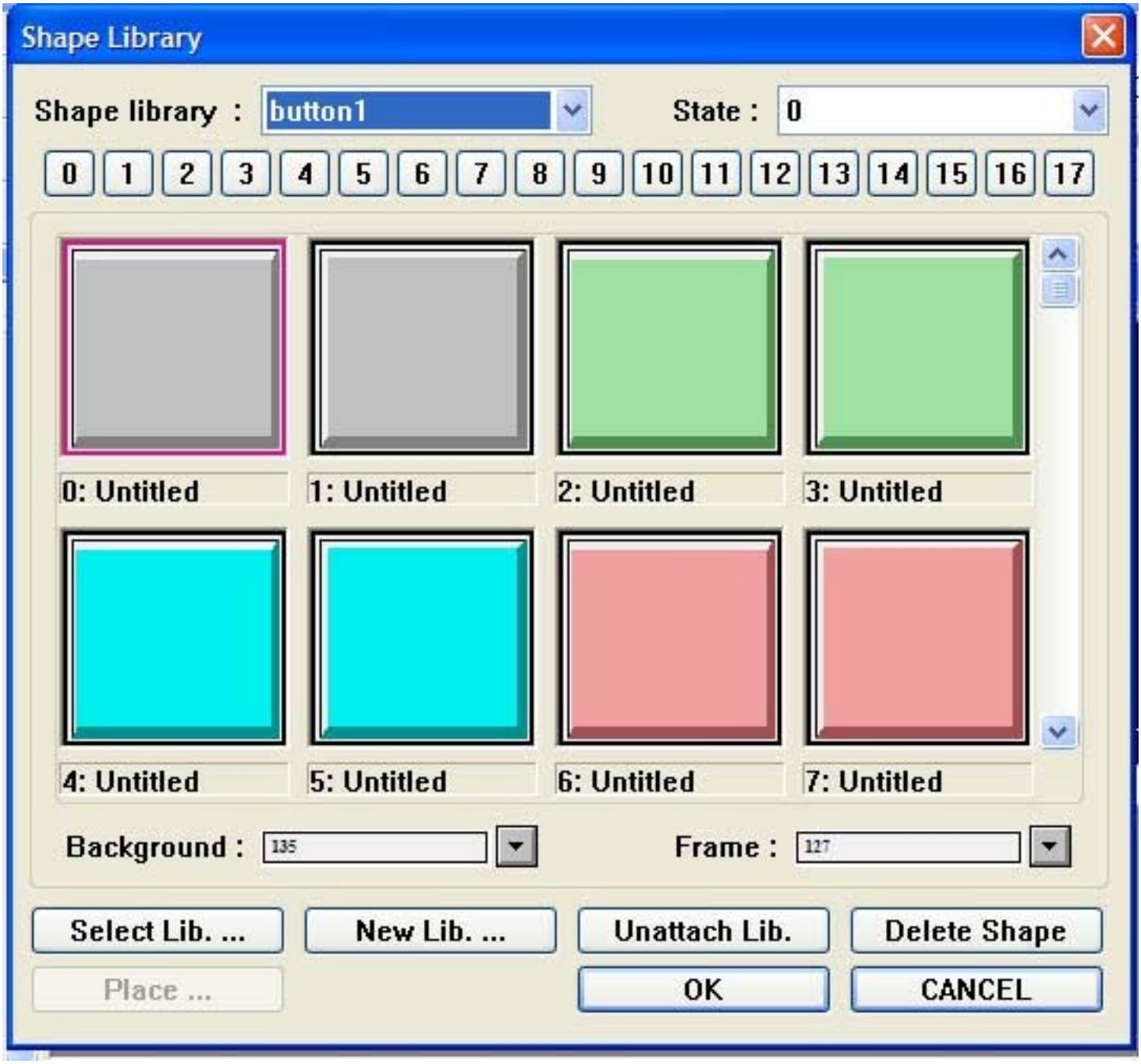
- balloons.slb(contains dialog boxes)
- captions.slb(contains various rectangle shapes)
- button1.slb(switches and lamps)
- button2.slb(switches and lamps)
- button3.slb(switches and lamps)
- button4.slb(switches and lamps)
- buttons.slb(switches and lamps)
- electric.slb(schematic symbols)
- isa_1.slb(isa symbols)
- isa_2.slb(isa symbols)
- part1.slb(switches and flags)
- pipe_m0.slb(pipes)
- pipe_m1.slb(pipes)
- pipes.slb(pipes)

When you start a new project, two libraries (button1 and button2) are automatically open.

To display the Shape Library dialog box 

1. From the **Library** menu, click **Shape**, then click **Call up library**. Or click the **Call up Shape Library** icon from the Standard toolbar. The Shape Library dialog box appears.

Let’s look at the Shape Library dialog box:



1. With the Shape Library dialog box displayed, click **Select Lib…** The Open dialog box appears.
2. Select which library you would like to open, then click **Open**. The Shape Library dialog box reappears with the selected library displayed.

If the project has ten libraries open, you may want to close a library so that another one may be opened.

To close or 'unattach' an open shape library

1. With the Shape Library dialog box displayed, click the pull-down box of **Shape library:** and choose from one of the open libraries of shapes.
2. Select which library you would like to close, then click **Unattach Lib.** A dialog box appears asking if you really want to unattach the selected library. Click **Yes**.
3. The Shape Library dialog box reappears and the library is removed from the **Shape library:** list.

You have the option of creating a new library. This can be very useful when organizing shapes.

To create a new shape library

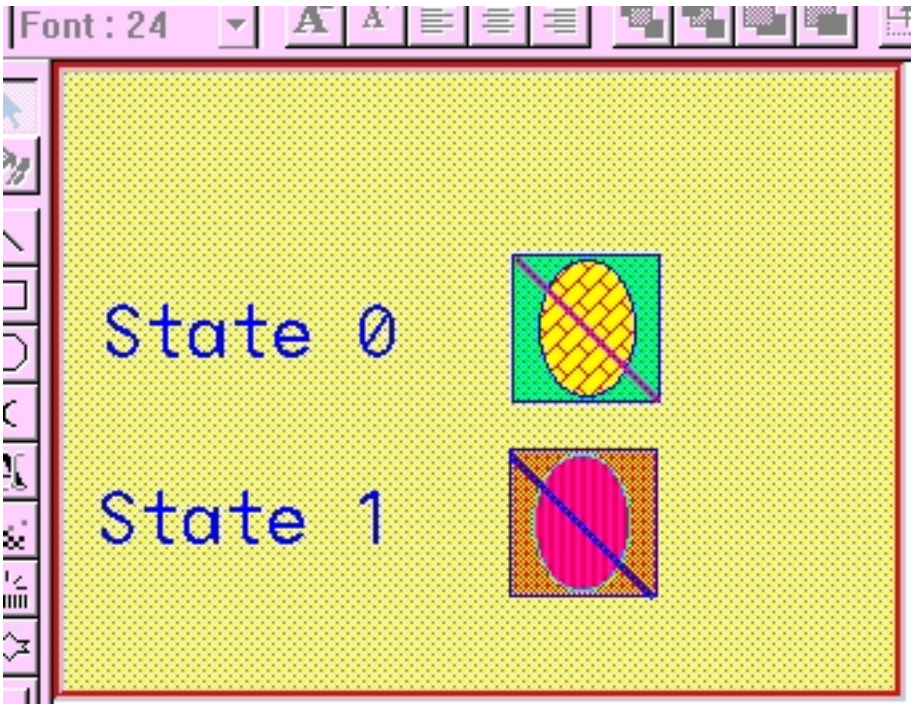
1. With the Shape Library dialog box displayed, click New Lib… The New Library dialog box appears.
2. Enter a name for the new library. Click **OK**.
3. The Shape Library dialog box appears with the new library loaded. Click **Close**.
4. The new library is now available to this project but it is not stored into the **Library** subdirectory of EasyBuilder until the project file has been saved.

Creating, deleting and placing shapes

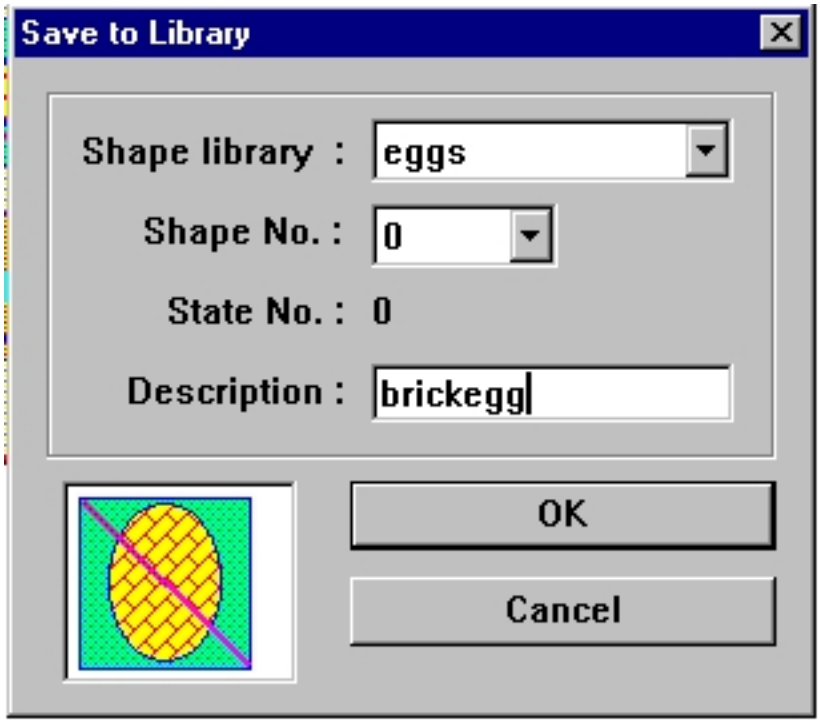
EasyBuilder provides drawing tools that can be used to create shapes. For more information on how to use the drawing tools, consult the first part of this chapter. The shapes that you create can be permanently stored into the shape libraries for access by any project.

To add a new shape to a library 

1. This example shows how to add a two-state shape to a library. You must first create the shapes that you plan on storing using the drawing tools supplied with EasyBuilder. The shapes can be any size. You can use any combination of drawing tools to create the shape.

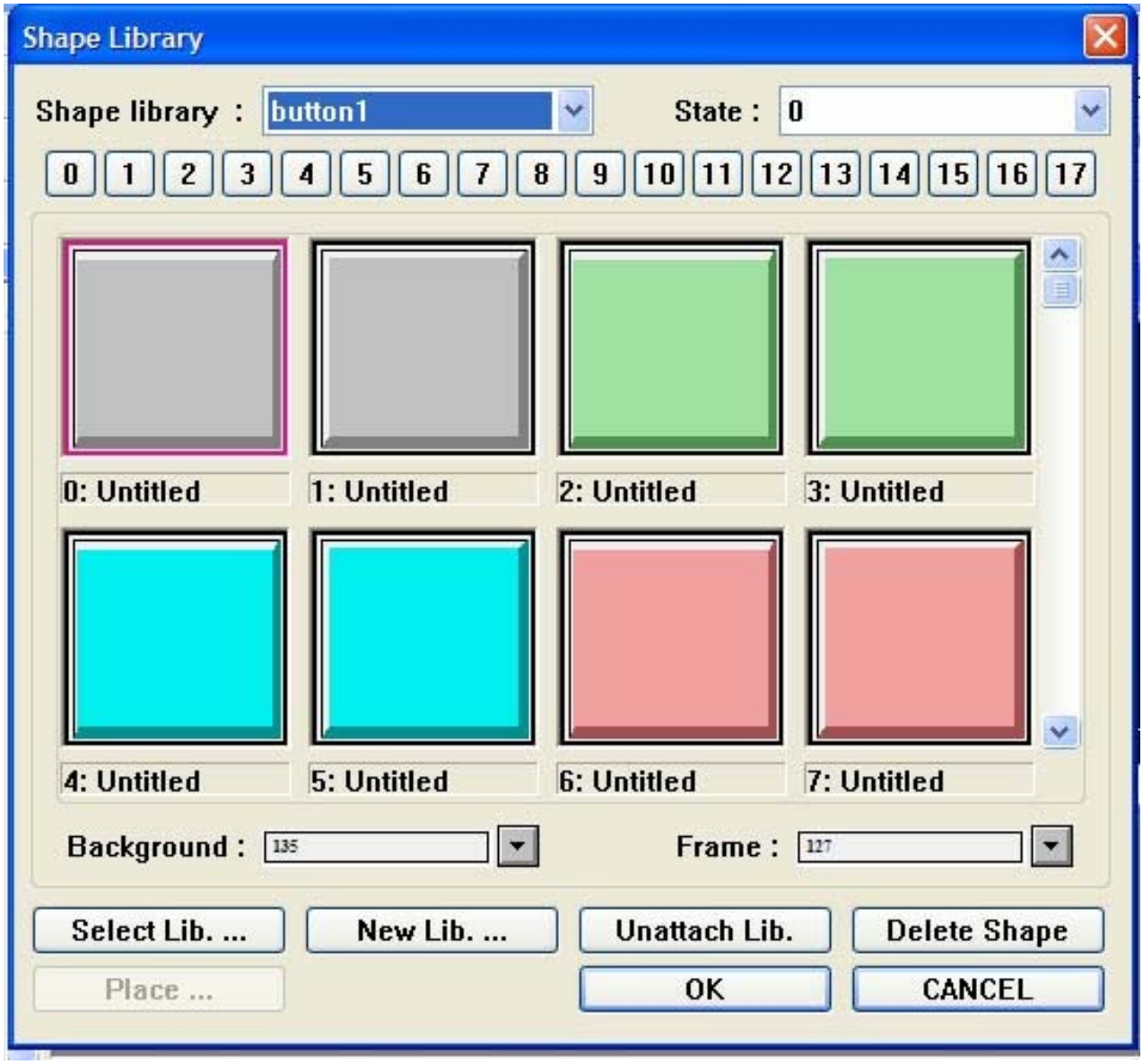


2. Highlight the shape by clicking the mouse and dragging a rectangular outline so that it covers the entire shape. When you are done, you should see small black squares around the perimeter of the shape.
3. From the **Library** menu click **Shape**, then **Save to library**. Or click the **Save Objects to Shape Library** icon in the Standard toolbar. The Save to Library dialog box appears.



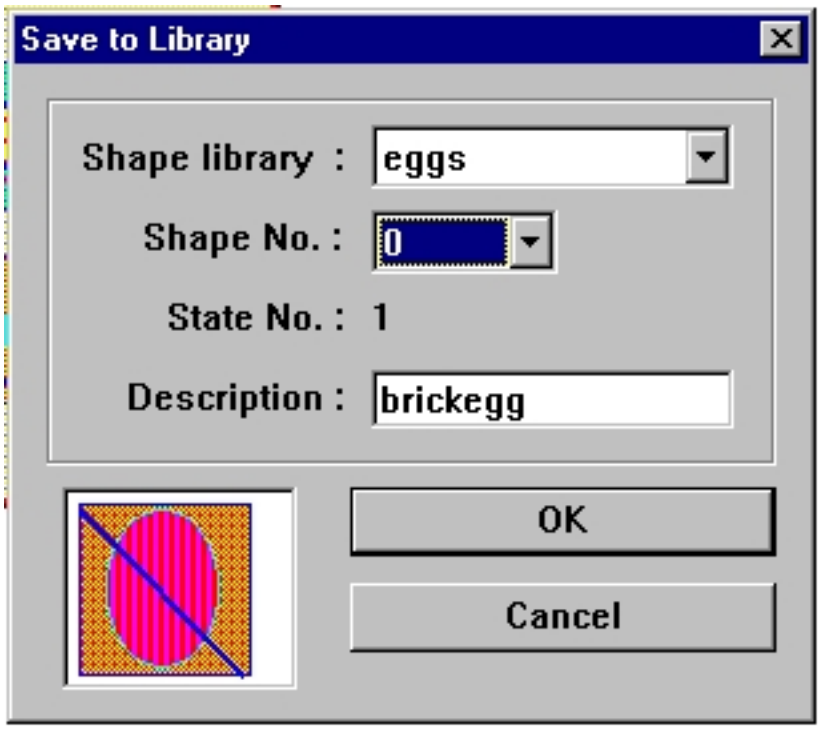
4. Click the **Shape library:** box and select the library you wish to store the shape in, (for this example **eggs**).
5. Click the **Shape No.:** pull-down box to select the location (0) to store the shape. Note that the **State No.:** will automatically change to **0**, first unused state.
6. Enter a title for the new shape in the **Description:** box.

7. Click **OK**. The Shape Library dialog box appears with the shape stored in the library.



8. Click **Close** to return to the main screen of EasyBuilder.
9. From the **File** menu, click **Save** to save the shape to the library. Important- you must save before attempting to store the next shape.
10. Highlight the other shape by clicking the mouse and dragging a rectangular outline so that it covers the entire shape.

11. Click the **Save Objects to Shape Library** icon in the Standard toolbar. The Save to Library dialog box appears.



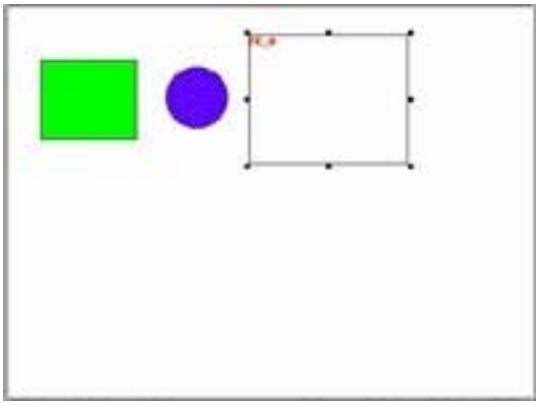
12. Click the **Shape library:** box and select the library you wish to store the shape in, (for this example **eggs**).
13. Click the **Shape No.:** pull-down box to select the location (0) to store the shape. Note that the **State No.:** will automatically change to state **1**, the next available state.
14. Click **OK**. The Shape Library dialog box appears with the shape stored in the library under **State 1**.
15. Click **Close** to return to the main screen of EasyBuilder.
16. From the **File** menu, click **Save** to save the shape to the library.

When adding new shapes to existing libraries that come with the EZware-500 software, remember that any future upgrade that you receive from Weintek may overwrite existing libraries during installation. This will delete any new shapes that you have entered! Therefore, we recommend that you create new libraries to store these shapes or make backup files of the libraries before installing new upgrades.

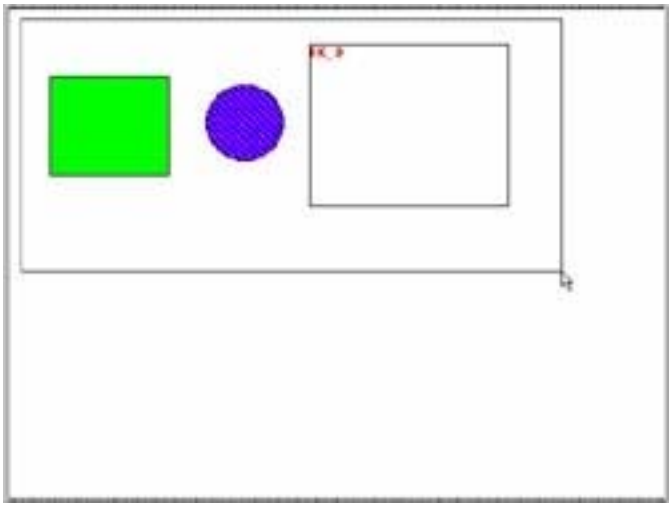
Another method of adding a new shape

1. When a shape is added to the library, the shape is placed at the upper, left-hand corner of the cell. In some cases, however, it might be necessary to place the shape in a particular location within the cell.
2. On the Window Property dialog (from the **Option** menu, select **Window Property**), check the **Using Function Key To Make Shape Library** checkbox.
3. Create the desired shape. Create a Function Key. The area defined by the Function Key will

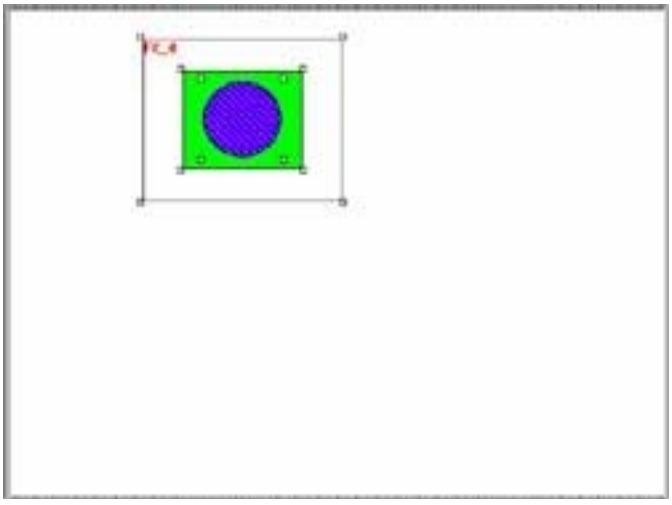
be the area that contains the shape. The mode of the Function Key does not matter. Do not assign a shape or a label. This example shows a shape placed in the center of the library area.



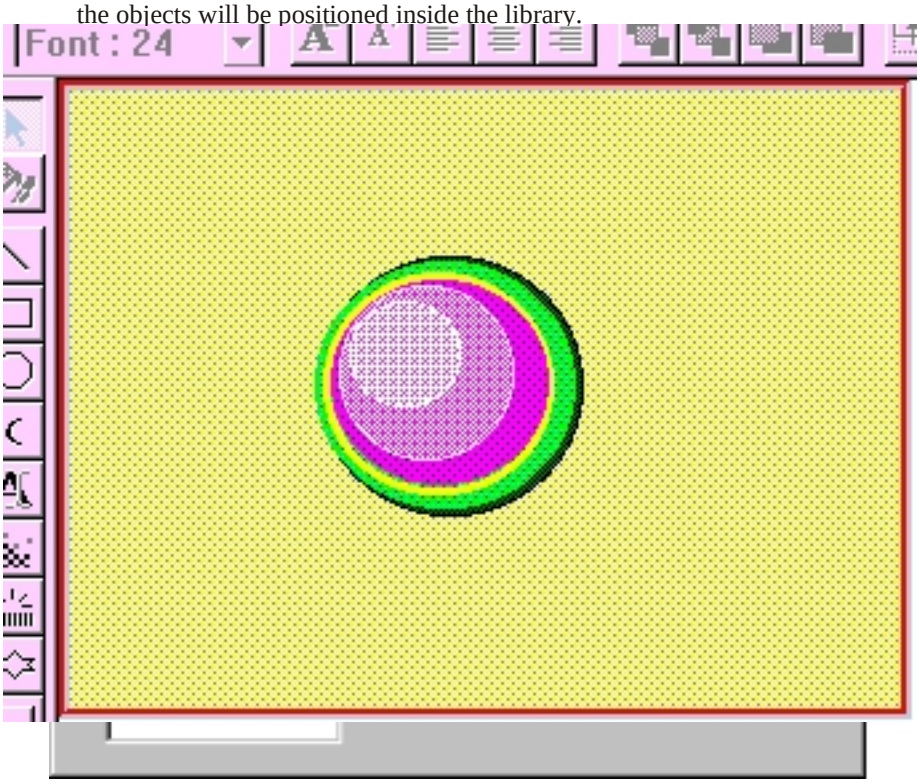
4. Position the objects that compose the new shape inside the function key.



5. Select all objects, including the Function Key.



6. From the **Library** menu, click **Shape**, then **Save to Library**. In the dialog box, notice how the shape is positioned. The location of the objects inside the Function Key area defines how the objects will be positioned inside the library.



If you find that some of the shapes included in the libraries are not useful, you have the option of deleting them so that they may be replaced by another shape.

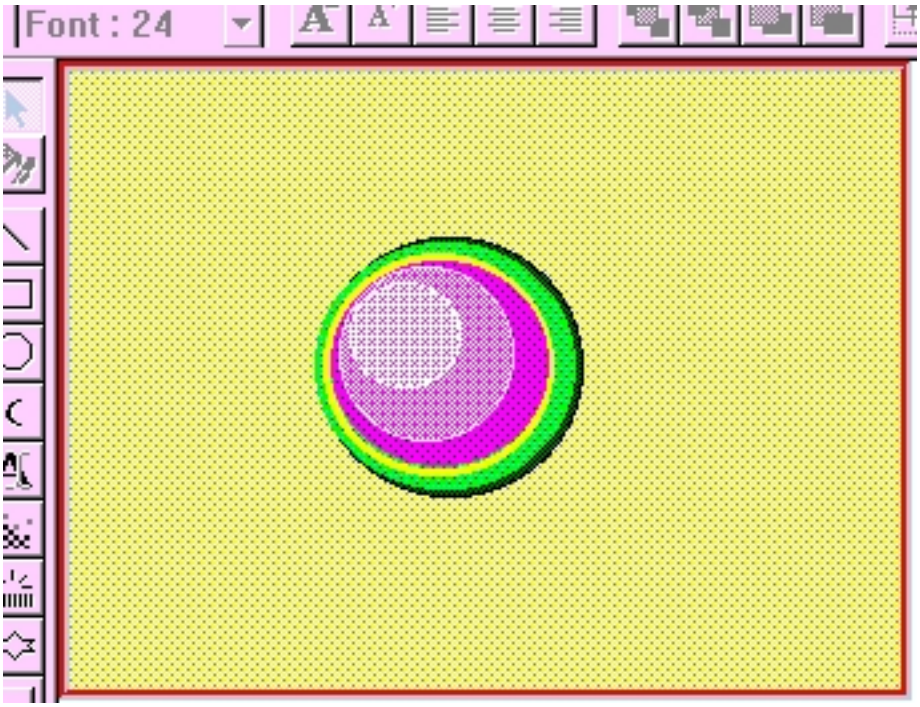
To delete a shape from a library

1. From the **Library** menu, click **Shape**, then **Call up library**.
2. Select the library that the shape is located in.
3. Click on the shape that you wish to delete from the library.
4. Click **Delete**. The shape is removed from the library.
5. Click **Close** to go back to the main screen of EasyBuilder.
6. Save the project file to permanently save changes made to the library. Caution – once project is saved, the shape is permanently deleted from the library and cannot be replaced.

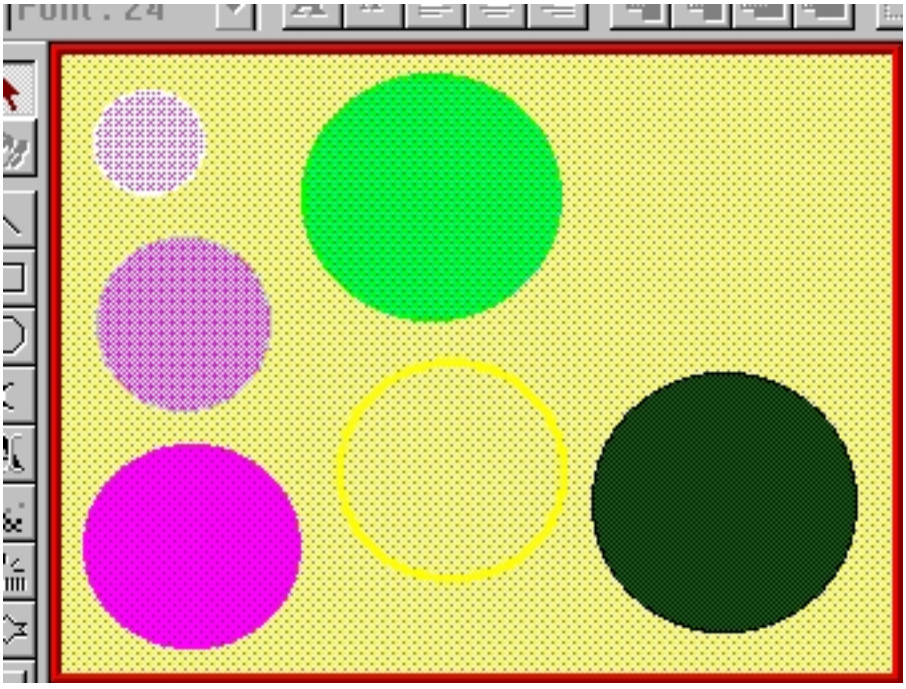
Finally, you may want to use all or parts of an existing shape to create a new one. To do this, you must first 'place' the shape onto the work area of EasyBuilder for editing.

To place a shape on the work area

1. From the **Library** menu, click **Shape**, then **Call up library**.
2. Select the library that the shape is located in. For this example, select **button1**.
3. Click on the shape that you want to place on the work area. For this example, select item **12**.
4. Click **Place…** The main screen of EasyBuilder reappears with the shape in the upper-left corner of the work area.
5. For this example, move the shape to the middle of the work area, as shown



6. Highlight the shape and separate each separate component.



7. As you can see, this particular shape is composed of six objects. You can edit these objects to change the shape or create a new shape from these objects. Please note that you are not actually changing the shape stored in the library. To change the shape, you must delete the shape and store any changes you make to a new shape.

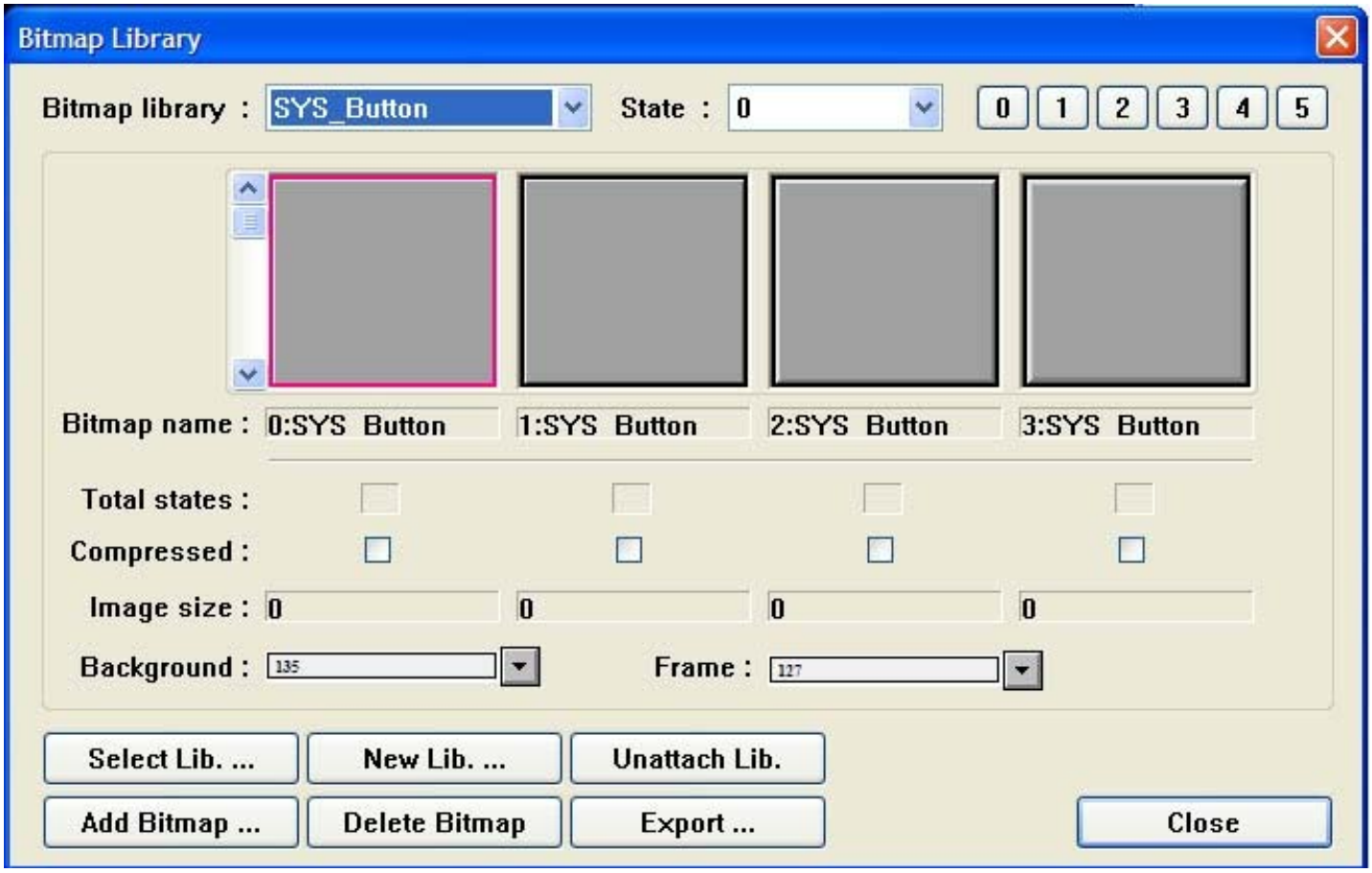
Using Bitmap Libraries

EasyBuilder includes six bitmap libraries that contain bitmaps you can select from:

- switch1.blb(buttons, lamps)
- bmp1.blb(miscellaneous)
- icons.blb(miscellaneous)
- pipline.blb(pipes)
- pictures.blb(various pictures)
- symbol1.blb(symbols)

To display the Bitmap Library dialog box 

1. From the **Library** menu, click **Bitmap**, then click **Call up library**. Or click the **Call up Bitmap Library** icon from the Standard toolbar. The Bitmap Library dialog box appears.



The Bitmap Library provides several options which are discussed in the following sections.

Accessing and Creating Bitmap Libraries

You can open, close, or even create new bitmap libraries easily using the Bitmap Library dialog box.

To open other bitmap libraries

1. With the Bitmap Library dialog box displayed, click **Select Lib…** The Open dialog box appears.
2. Select which library you would like to open, then click **Open**. The Bitmap Library dialog box reappears with the selected library displayed.

If the project has ten libraries open, you may want to close a library so that another one may be opened.

To close or 'unattach' an open bitmap library

1. With the Bitmap Library dialog box displayed, click the pull-down box of **Bitmap library:** and choose from one of the open libraries of bitmaps.
2. Select which library you would like to close, then click **Unattach Lib.** A dialog box appears asking if you really want to unattach the selected library. Click **Yes**.
3. The Bitmap Library dialog box reappears and the library is removed from the **Bitmap library:** list.

You have the option of creating a new library. This can be very useful when organizing bitmaps.

To create a new bitmap library

1. With the Bitmap Library dialog box displayed, click New Lib… The New Library dialog box appears.
2. Enter a name for the new library. Click **OK**.
3. The Bitmap Library dialog box appears with the new library loaded. Click **Close**.
4. The new library is now available to this project but it is not stored into the **Library** subdirectory of EasyBuilder until the project file has been saved.

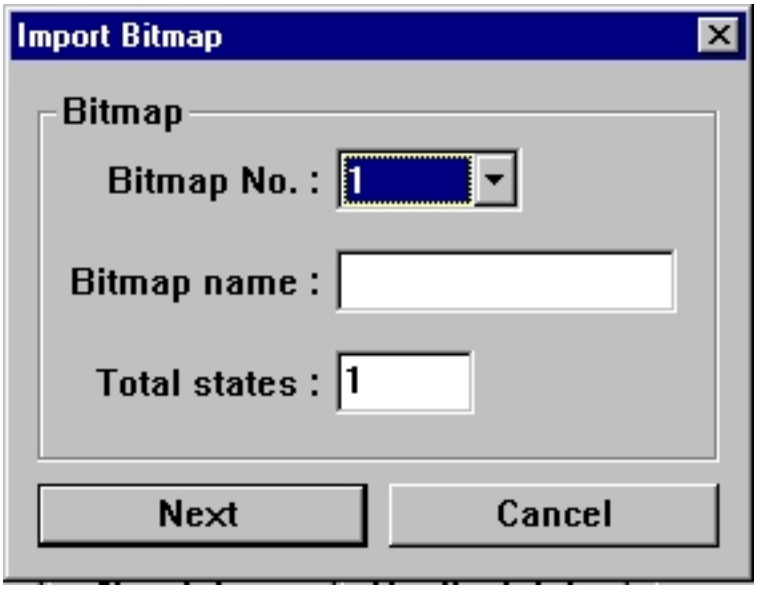
Creating and exporting bitmaps

Although EasyBuilder does not provide the ability to create bitmaps, the software can import bitmaps created from some other application program. This allows the ability to use a wide selection of bitmaps that can be permanently added to your bitmap libraries. EasyBuilder only accepts bitmaps that are 256 color or less resolution.

To add a new bitmap to a library

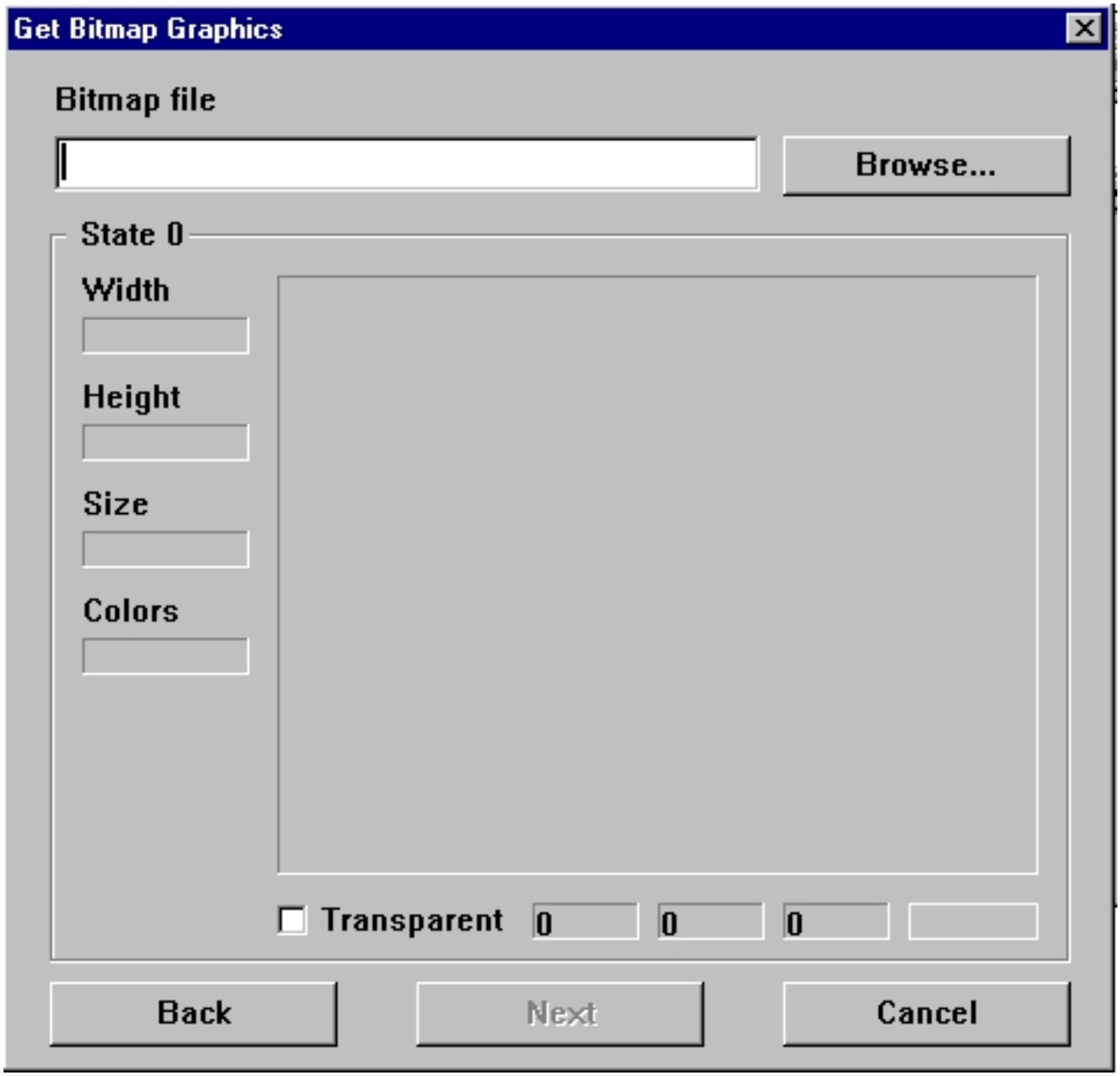
1. Display the Bitmap Library dialog box.
2. Click the **Bitmap Library:** pull-down box and select one of the bitmap libraries.
3. Click on the box location where you would like to enter the new bitmap.

4. Click **Add Bitmap…** The Import Bitmap dialog box displays.



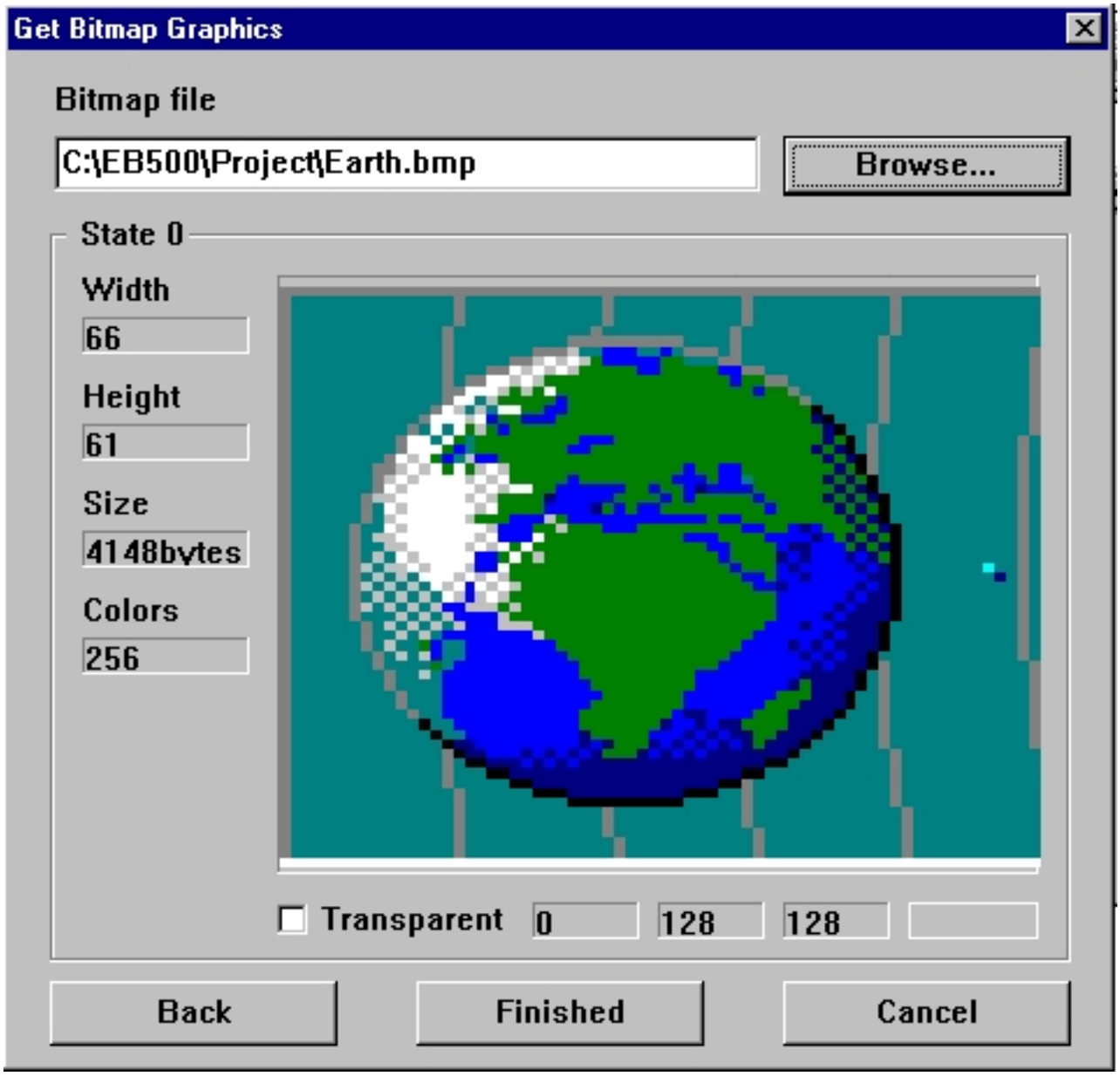
5. The **Bitmap No.:** is the location of the bitmap in the bitmap file. This should reflect the box location that you selected in step 4.
6. Enter a title for the new bitmap in the **Bitmap name:** box.
7. Enter the total number of states (max. 32) for the bitmap in the **Total states:** box.

8. Click **Next**. The Get Bitmap Graphics dialog box appears.



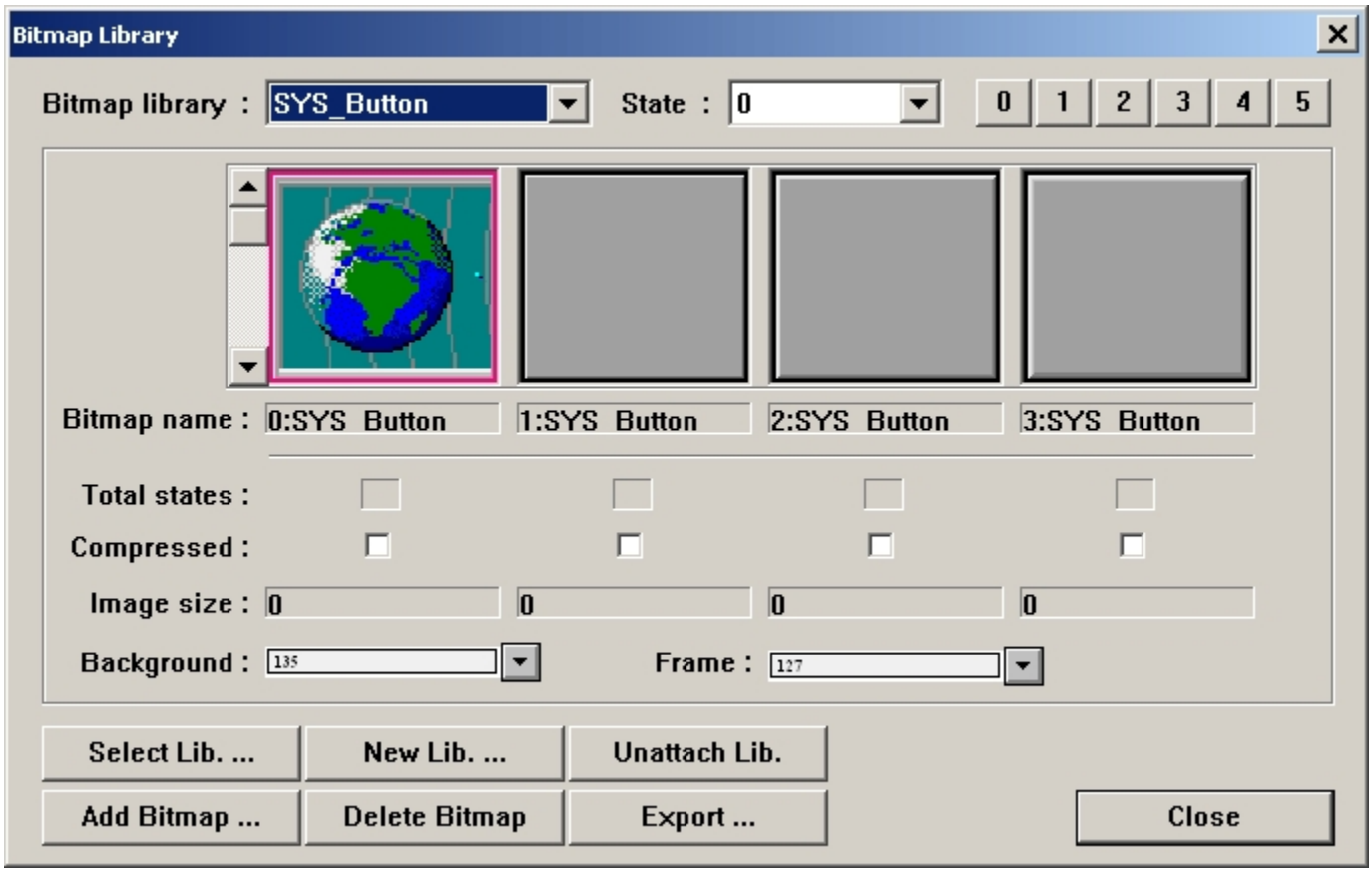
9. Click **Browse** to search for the bitmap. The Open dialog box appears. Select the bitmap file and click **Open**.

10. The Get Bitmap Graphics dialog box reappears with the bitmap displayed, along with information about the bitmap.



11. The **Transparent** checkbox is used to make any color selected on the bitmap 'transparent'. If the color is configured as transparent, then when the bitmap is placed on a window, all areas of the bitmap with the transparent color will not be shown. This is very useful when you want to overlap several bitmaps, (for example, when configuring a bar graph). To use the transparent feature, click on the bitmap picture in the Get Bitmap Graphics dialog box on the color that is to be transparent. The three boxes to the right of the **Transparent** checkbox will show the red, green, and blue mixture of the color and the fourth box actually shows the color. Then check the **Transparent** checkbox.
12. If you are creating a multi-state bitmap, then click **Next** to select the next bitmap picture for

the other states. When the last state is completed, then click **Finished**. The Bitmap Library dialog box reappears with the bitmap listed.



13. You have the option of selecting **Compressed** format for any of the bitmaps in a library. The compressed option decreases the size of the bitmap thereby reducing the memory requirements of the HMI to store that bitmap. However, because it is compressed, the HMI will require slightly more processing time to display the bitmap. Speed is sacrificed for memory. We recommend that you do not use the compression option except in cases where you are reaching the maximum limits of storage capability of the HMI.
14. Click **OK** to return to Create Bitmap Object. Click **OK** again to return to the main screen of EasyBuilder. Place the bitmap object somewhere in the work area.
15. From the **File** menu, click **Save** to permanently save the bitmap to the library.

When adding new bitmaps to existing libraries that come with the EZware-500 software, remember that any future upgrade that you receive from Weintek may overwrite existing libraries during installation. This will delete any new bitmaps that you have entered! Therefore, we recommend that you create new libraries to store these bitmaps or make backup files of the libraries before installing new upgrades.

If you find that some of the bitmaps included in the libraries are not useful, you have the option of deleting them so that they may be replaced by another bitmap.

To delete a bitmap from a library

1. From the **Library** menu, click **Bitmap**, then **Call up library**. The Bitmap Library dialog box appears.
2. Select the library that the bitmap is located in.
3. Click on the bitmap that you wish to delete from the library.

4. Click **Delete bitmap**. The bitmap is removed from the library.
5. Click **Close** to go back to the main screen of EasyBuilder.
6. Save the project file to permanently save changes made to the library. Caution – once the project is saved, the bitmap is permanently deleted from the library.

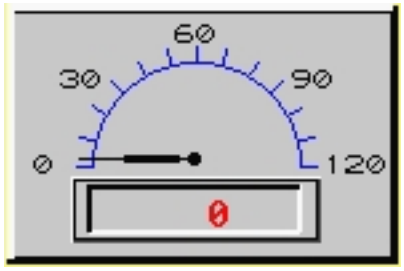
Finally, you may want to export a bitmap located in one of the bitmap libraries so that it can be modified or used in other programs.

To export a bitmap

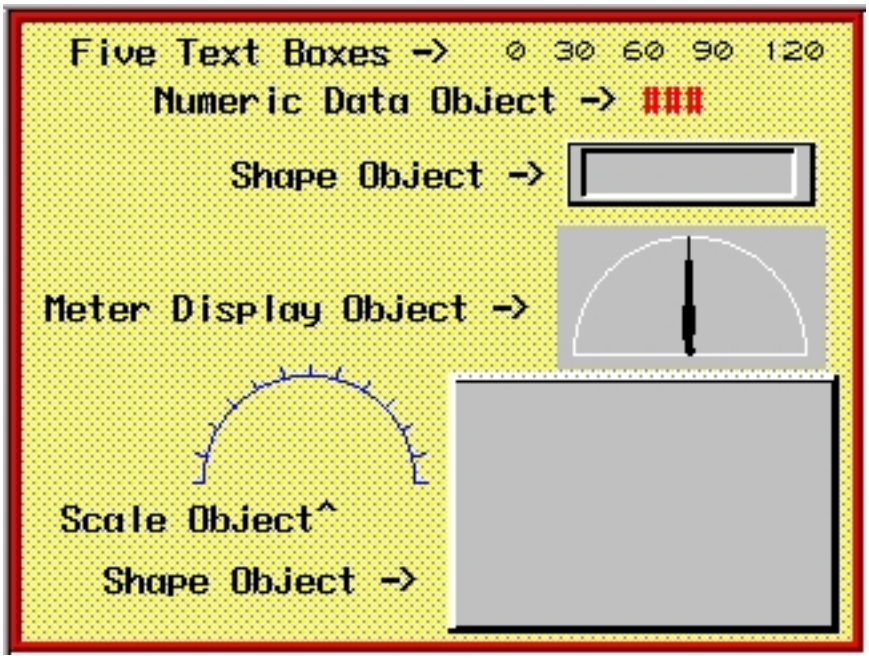
1. From the **Library** menu, click **Bitmap**, then **Call up library**. The Bitmap Library dialog box appears.
2. Select the library that the bitmap is located in.
3. Click on the shape that you want to export.
4. Click **Export…** The Save As dialog box appears.
5. In the **File name:** box, enter a file name to store the bitmap in.
6. Click **Save**. The bitmap is saved and the Bitmap Library reappears.
7. Click **Cancel** to exit the Bitmap Library and **Cancel** again to exit the Create Bitmap Object dialog box.

Using Group Libraries

Any graphics object or objects displayed on the work area of EasyBuilder can be grouped together and stored into a group library. Graphics objects can be passive or active. The groups that you create can be permanently stored into the group libraries for access by any project. Creating group objects often saves a lot of time when creating new projects because you can use the same objects repeatedly. For example, suppose you regularly use a scale meter similar to the following:



This one scale meter is actually composed of ten objects.



Rather than waste time recreating this scale meter for each new application, it would be much easier to store the meter into a group library.

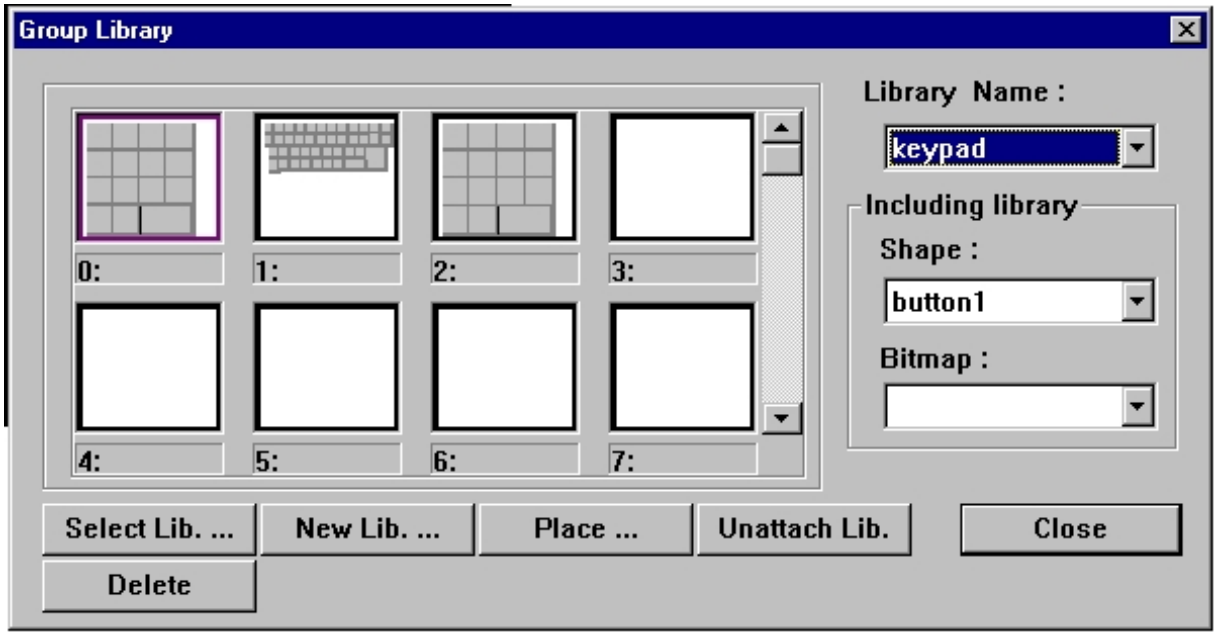
EasyBuilder includes one group library (keypad.glb) that contains sample keypads you can select from.

When a new project is started, the keypad group library is automatically open.

To display the Group Library dialog box 

1. From the **Library** menu, click **Group**, then click **Call up library**. Or click the **Call up Group Library** icon from the Standard toolbar. The Group Library dialog box appears.

Let’s look at the Group Library dialog box:



The Group Library provides several options which are discussed in the following sections.

Accessing and Creating Group Libraries

Group libraries can be opened and closed easily using the Group Library dialog box. It is even possible to create new group libraries.

To open other group libraries

1. With the Group Library dialog box displayed, click **Select Lib…** The Open dialog box appears.
2. Select which library you would like to open, then click **Open**. The Group Library dialog box reappears with the selected library displayed.

If the project has ten libraries open, you may want to close a library so that another one may be opened.

To close or ‘unattach’ an open group library

1. With the Group Library dialog box displayed, click the pull-down box of **Library Name:** and choose from one of the open libraries of groups.
2. Select which library you would like to close, then click **Unattach Lib.** A dialog box appears asking if you really want to unattach the selected library. Click **Yes**.
3. The Group Library dialog box reappears and the library is removed from the **Group library:** list.

The option of creating a new library is also available. This can be very useful when organizing groups.

To create a new group library

1. With the Group Library dialog box displayed, click **New Lib…** The New Library dialog box appears.
2. Enter a name for the new library. Click **OK**.

3. The Group Library dialog box appears with the new library loaded. Click **Close**.
4. The new library is now available to this project but it is not stored into the **Library** subdirectory of EasyBuilder until the project file has been saved.

Creating, deleting and placing groups

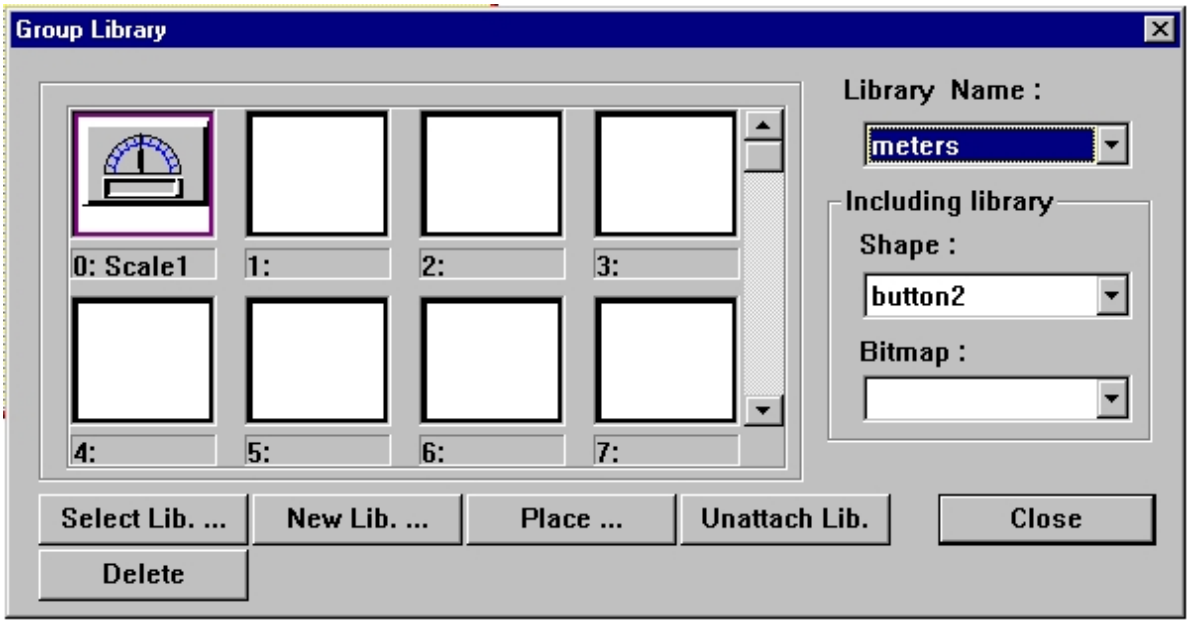
Once a complex group object has been created, it must be stored permanently into a group library where it can be retrieved and used in multiple projects. At any time, it is possible to place a group object onto the work area of EasyBuilder for usage or editing.

To add a new group to a library

1. Highlight the group object by clicking the mouse and dragging a rectangular outline so that it covers the entire group. When you are done, you should see small black squares around the perimeter of the group. For this example, the scale meter shown at the beginning of this section is highlighted.
2. From the **Library** menu click **Group**, then **Save to library**. Or click the **Save to Group Library** icon in the Standard toolbar. The Save to Library dialog box appears.

3. Click the **Group library:** box and select the library you wish to store the group in, (for this example **meters**).
4. Click the **Group No.:** pull-down box to select the location (0) to store the group. Note that the **Group No.:** will automatically change to **0**, (first group not used).
5. Enter a title for the new group in the **Description:** box, (ex. Scale1).

6. Click **OK**. The Group Library dialog box appears with the group stored in the library.



7. Notice that any shape or bitmap libraries used to create the group object are listed for your reference in the **Including library** box.
8. Click **Close** to return to the main screen of EasyBuilder.
9. From the **File** menu, click **Save** to permanently save the group to the library.

When adding new group objects to existing group libraries that come with the EZware-500 software, remember that any future upgrade that you receive from Weintek may overwrite existing libraries during installation. This will delete any new group objects that you have entered! Therefore, we recommend that you create new libraries to store these group objects or make backup files of the libraries before installing new upgrades.

If you find that some of the groups included in the libraries are not useful, you have the option of deleting them so that they may be replaced by another group.

To delete a group from a library

1. From the **Library** menu, click **Group**, then **Call up library**.
2. Select the library that the group is located in.
3. Click on the group that you wish to delete from the library.
4. Click **Delete**. The group is removed from the library.
5. Click **Close** to go back to the main screen of EasyBuilder.
6. Save the project file to permanently save changes made to the library. Caution – once a project is saved, the group is permanently deleted from the library and cannot be replaced.

Finally, to place a group object onto the work area for use or editing, do the following.

To place a group on the work area

1. From the **Library** menu, click **Group**, then **Call up library**.
2. Select the library that the group is located in. For this example, select **keypad**.
3. Click on the group that you want to place on the work area. For this example, select item **0**.

4. Click **Place…** The main screen of EasyBuilder reappears with the group in the upper-left corner of the work area.
5. Move the group object to the preferred location on the window.

This completes our discussion on creating shapes, bitmaps, and group objects with EasyBuilder. This chapter also provided some examples on how to use the drawing tools. The next several chapters describe the many parts or active graphics objects that can be placed onto windows.

Chapter 7 - Creating and Using Databases and Languages

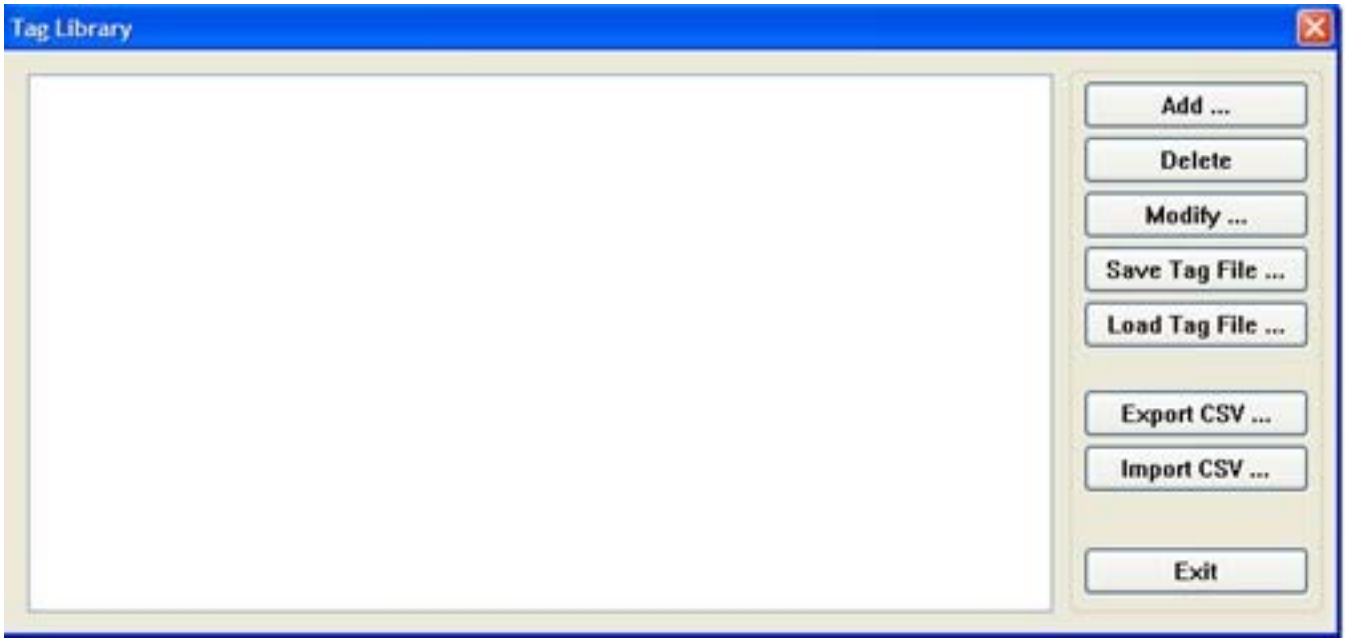
EZBuilder includes the ability to create and use both a Tag Database Library and a Label Database Library. The Tag Library is a database of PLC register addresses. Once created, the Tag Database allows individual object address assigning through Tag selection from the Tag Library. The Label Library is a database of text labels. Once created, the Label Database allows individual object text labeling through Label selection from the Label Library. The Label text may also be represented in up to 4 languages, each selectable for display.

Creating and Using the Tag Library

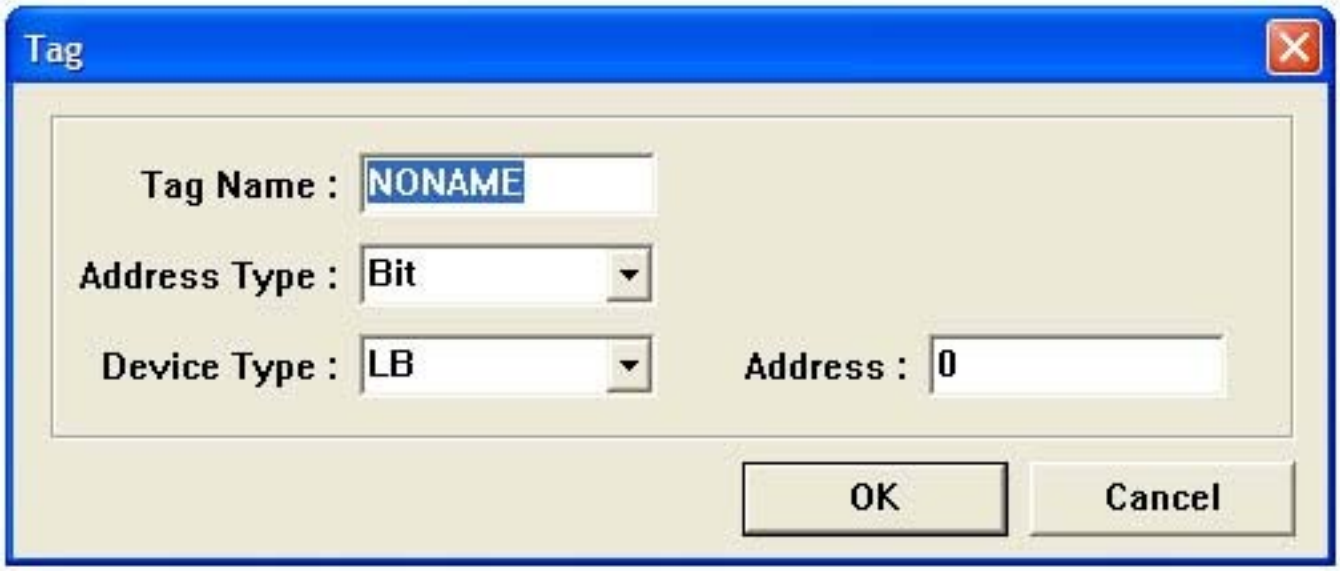
The Tag Library is a database of register addresses, with each tag representing a single address. The Tag listing does not contain display information, as display attributes are set when the individual object using the Tag listing is created.

▶ To create a Tag Database Library 

1. From the **Library** menu, click **Tag...**, or click the Call Up Tag Library icon from the Standard toolbar. The Tag Library dialog box appears.



2. To create a new Tag listing, click the **Add** button to display the Tag dialog.



3. Enter the new *Tag name*. Tag names may be up to 100 characters, any character.
4. Enter the *Address type* (Bit or Word).
5. Enter the *Device type*.
6. Enter the *Register Address.*
7. Make additional entries to the Database by repeating the procedure.

To delete a tag:

1. Open the Tag Library as directed above.
2. Select the tag to delete.
3. Click the **Delete** button.

To modify a tag:

1. Open the Tag Library as directed above.
2. Select the tag to edit.
3. Click the **Modify** button.
4. Edit as applicable and then click **OK**.

Importing and Exporting the Tag Library

This feature allows tags from the tag library to be saved into a .tgl file format. Once saved, the file can be loaded into another project.

To save the tag library:

1. Click on **Save Tag File...** The Open dialog box appears.
2. Enter the *name* of the file where you wish to save the data.
3. Click **Open**.

To load the Tag Library from an existing .tgl file:

1. Click on the **Load Tag File...**The Open dialog box appears.
2. Browse for the .tgl file that contains the tags.

3. Click **Open**.

The .tgl file format can not be edited.

The tag library can also be imported and exported using a comma-separated variable (.csv) file format. This format is very useful if you need to make several changes to the database; or, if creating a new project, it is easier and faster to create the tags in a .csv file.

To export the tag library to a .csv file:

1. Click **Export CSV**. The Open dialog box appears.
2. Enter the *name of the file* where you wish to save the data.
3. Click **Open**.

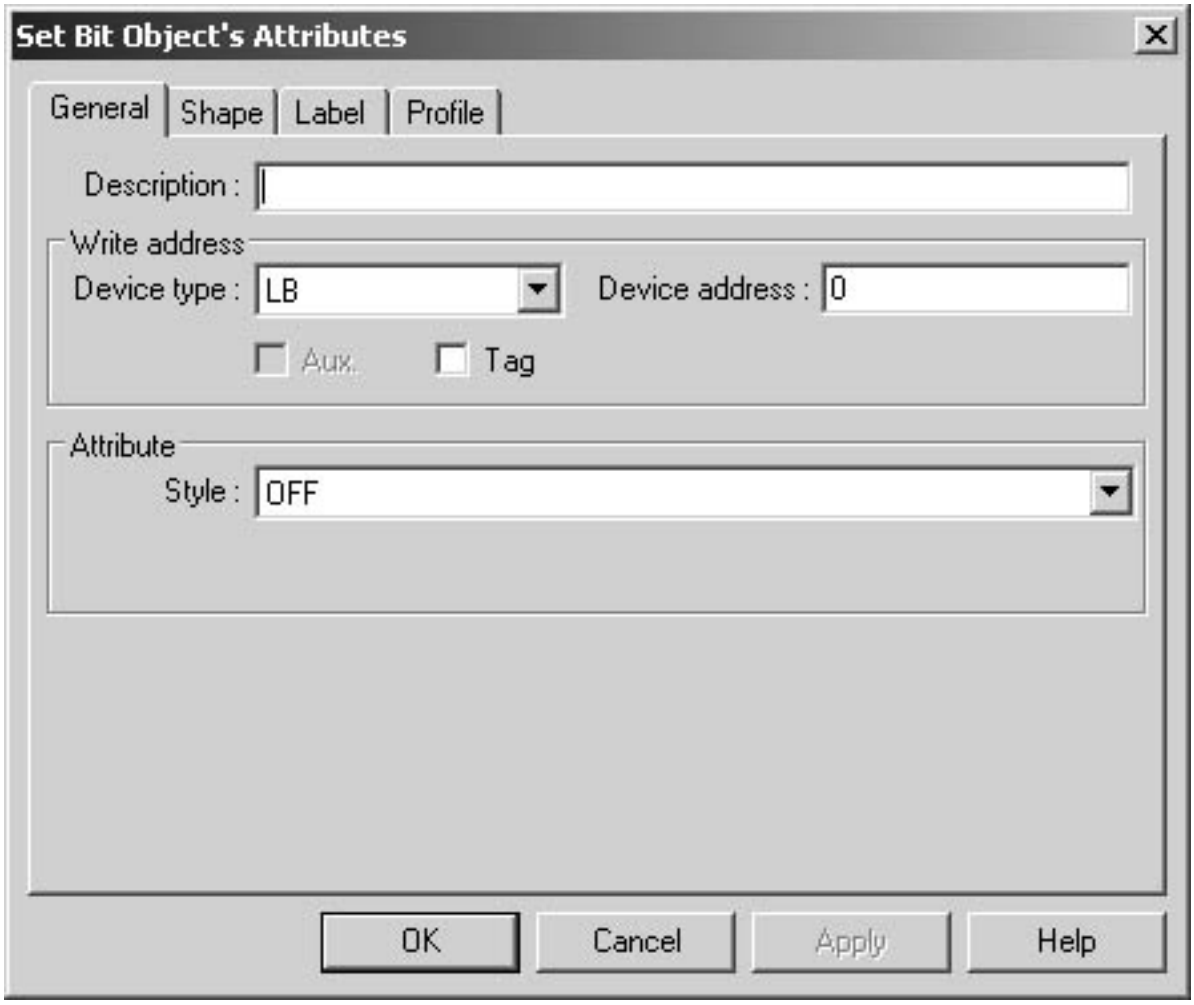
To import the tag library from a .csv file:

1. Click **Import CSV…** The open dialog box appears.
2. Brows for the CSV file containing the tags.
3. Click **Open**.

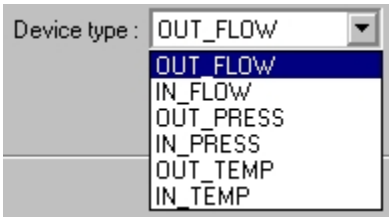
Using the Tag Library

Once the Tag Database has been created, individual objects can use the Library to assign a Tag to the object, referenced to the PLC register address.

1. On addressable objects, once the Tag Database has been created, a ***Tag Checkbox*** is available. This box is not available if no tags have been configured in the Database Library.



2. Check **Tag Checkbox**. The **Device Type** list will be filled with a list of available tags. Objects addressable as Register-type objects will display Tags configured as Word; Objects addressable as Bit-type objects will display Tags configured as Bit.



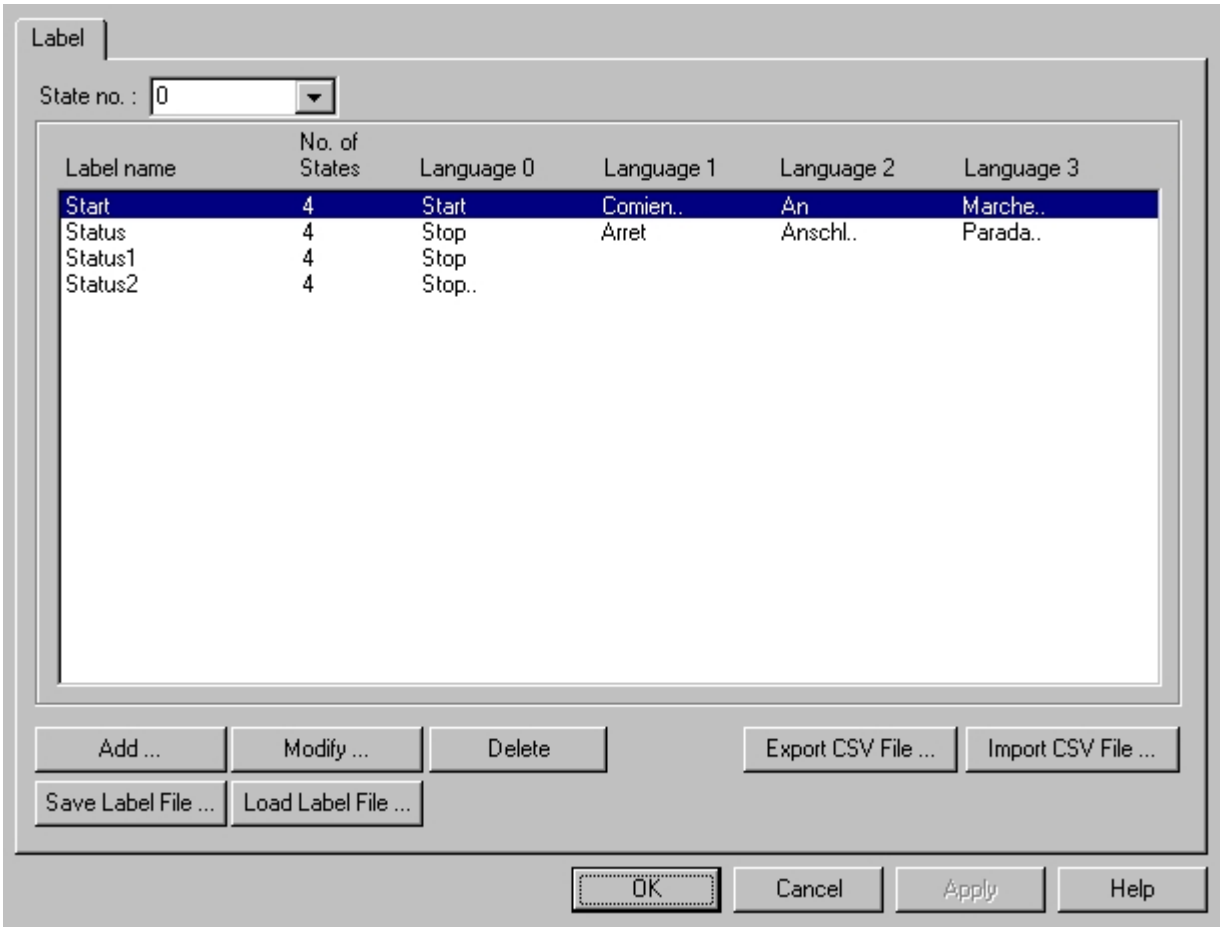
3. Select the desired *Tag* from the Device Type list to address the Object to the referenced Register Address.

Creating the Label Library

The Label library is a database of text Labels, for use with objects utilizing text labels descriptive of their status or condition. Each label can consist of up to 32 unique text strings (representing the maximum number of states available per object), displayable in up to four separate languages.

To create a label text database library

1. From the **Library** menu, click **Label...**, or click the callup label library icon from the standard toolbar. The Label Library dialog box appears.



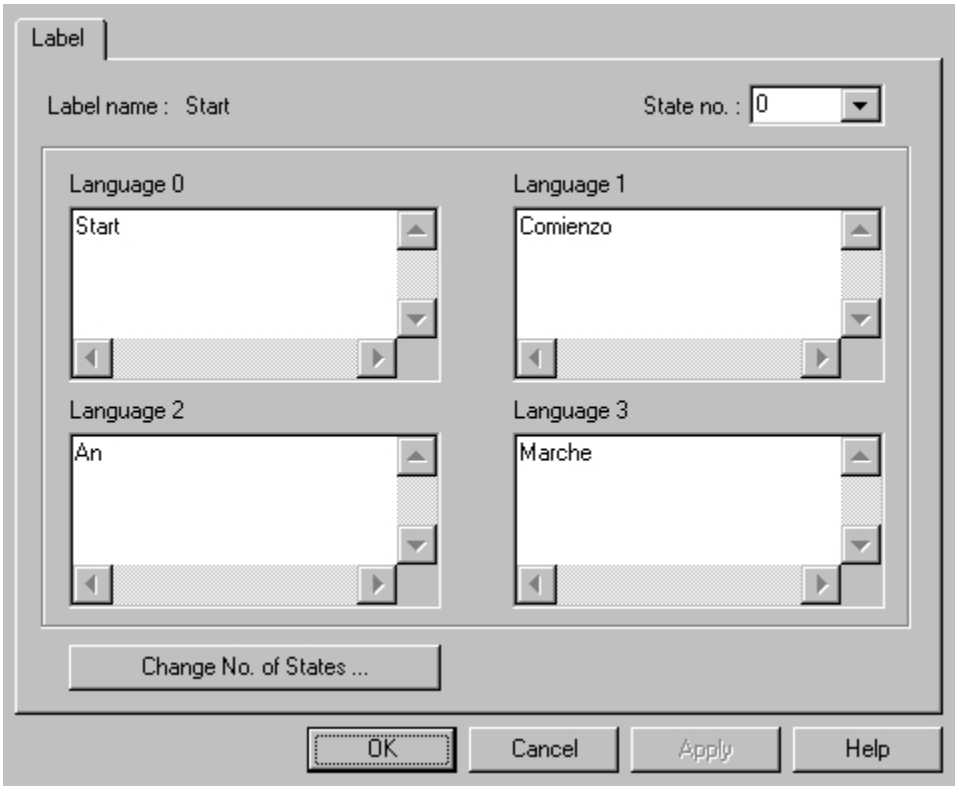
2. Create a label by opening the Label Library as detailed above and Clicking the **Add** button to display the Label dialog.

3. Enter the *Title (Name)* for the label, and how many states the new label requires. For example, a label Titled “Start” requiring 2 states, to represent separate conditions, where condition one status is “START”, condition two is “STOP”.

4. Click the **OK** button, and the new label is added to the database. The Name will appear in a selectable listing of available Labels for use with objects that support text field labeling.

5. The new label is still selected. Click the **Modify** button to display the ***Label Content Setting*** dialog. In the ***State No***. list pulldown box, select the state that the text is representing.

6. In each **Language** box, as desired, enter the text representing that state in each language. For example, "START" in Language 0 to represent English, and the appropriate equivalent wording in all other languages configured.



7. If further text labeling is required to represent conditions or status in additional states, while still in the ***Modify*** mode for Label Content Setting, select the state requiring text labeling. Enter the text in the Language boxes as above.

There is no inter-language conversion or translation capability. Entries must be made using the correct wording for the language designated, to display as entered.

8. Continue to add state descriptive text labeling as required.
9. When all text has been entered for all states desired, Click **OK.** The Label is now contained in the Label Library and displayed when the Library is opened. View text labels for the various states of each label in the Library by selecting the State to display in the pull down box.

10. Once a label text has been created and you wish to change the number of states, you may do so by double clicking on the desired label text. On the Label Content Setting dialog box, click the **Change No of States...**button, and then enter the *No. of states* desired for the label.

Importing and Exporting the Label Library

This feature allows you to save the labels from the label library into a .lbl file format. Once saved, the file can be loaded into another project.

To save the label library:

1. Click on **Save Label File...** The open dialog box appears.
2. Enter the *name of the file* where you wish to save the data.
3. Click **Open.**

To load the label library from an existing .lbl file:

1. Click **Load Label File...** The Open dialog box appears.
2. Browse for the lbl file that contains the labels.
3. Click **Open**.

The .lbl file format can not be edited.

The label library can also be imported and exported using a comma-separated variable (.csv) file format. This format is very useful if you need to make several changes to the database. When creating a new project, it is easier and faster to create tags as .csv files.

To export the label library:

1. Click Export **CSV File…** The Open dialog box appears.
2. Enter the *name of the file* where you wish to save the data.
3. Click **Open.**

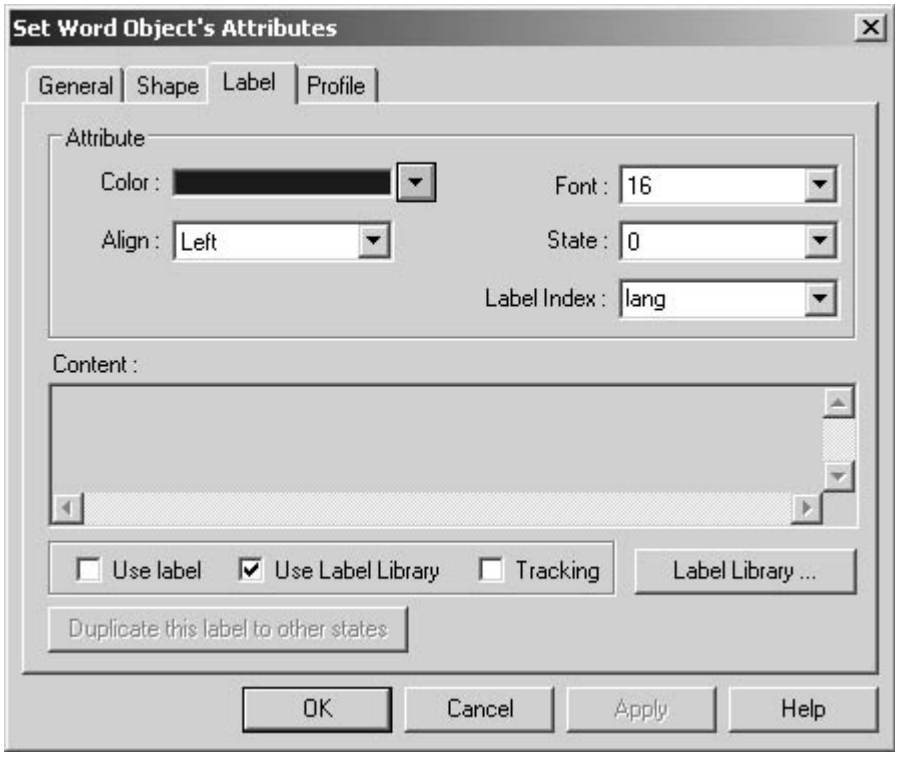
To import the label library:

1. Click **Import CSV File…** The Open dialog box appears.
2. Browse for the CSV file that contains the labels.
3. Click **Open**.

Using the Label Library

Once entries have been made to the Label Library, they can be used with any object that supports text field labeling.

1. Select an object that supports text field labeling as required. On the ***Label*** tab for the object's ***Attributes,*** check the **Use Label Library** checkbox. The Label Index list will be enabled, containing all of the labels in the database.



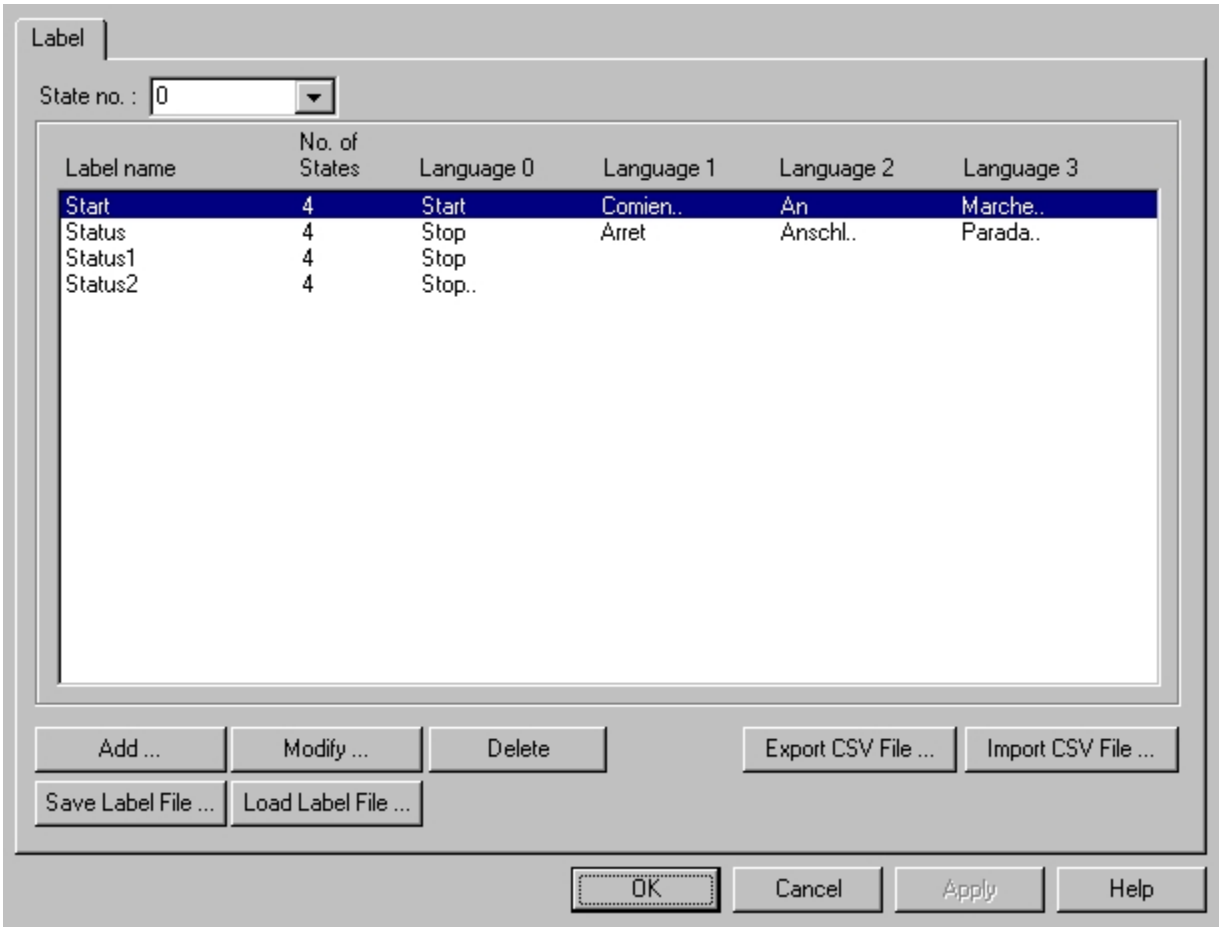
2. Select the applicable label from the list. The label will display on the object, representative for states and languages as entered in the ***Label Library*** dialog.
3. Once created with Labels assigned, objects can be viewed displaying the various text labeling in each of the states detailed, in each of the languages utilized. To view, on EZBuilder's Standard Toolbar, select the state and language for objects displayed on the window. The object's text labeling display will change according to the state and language selected for viewing.

4. To use the Multi-Language features of the Label Library to display object labels in languages other than the default language (Language 0), the function must be enabled. Enable the Multi-Language capability as detailed below.

Using Languages with the Label Library

The MT5xx supports up to 4 languages for use with the Label Library text labeling. In order to use the Multi-Language features, a label configured in multiple languages must be created and stored in the Label Library, as detailed above. The language displayed is selected through Internal Local Word Control, LW9130.

1. Text Labels in any of up to 4 languages can be displayed, as configured in the Label Library. In LW9130, a Value of 0 enables Language 0 as configured in the Label Library, a Value of 1 enables Language 1, a Value of 2 enables Language 2, and a Value of 3 enables Language 3.



2. The appropriate value must be written to LW9130 to enable a designated language to display. For example, a series of set word objects writing value to LW9130 could be placed on a setup screen. Each set word might be labeled with the Language to be enabled, and the appropriate

value addressed to LW9130. If Set Word #0 enables English and English is the configured Language 0 in the Label Library, pressing the English Set Word object writing a 0 to LW9130 will enable label text display in English. If Set Word #1 enables French and French is the configured Language 1 in the Label Library, pressing the French Set Word object writing a 1 to LW9130 will enable text display in French. Continue for all Languages configured in the Label Library.

Chapter 8 - Representing Data with Graphics Objects

syBuilder includes several active graphics objects or 'parts' which are used to represent data that is stored in the PLC internal memory of the HMI. The data represented can be single bit coils, 16-bit, 32-bit, or 64-bit registers. The data can be represented as numbers, ASCII characters, or as graphic shapes or bitmaps. This chapter focuses on only parts that perform relatively simple functions. More complex parts used for alarms, trending, etc. will be reserved for later chapters.

Using Internal Data Memory of HMI

The MT5xx contains internal data memory, which can be used to store information that either must be sent to or received from the PLC. The HMI has two types of Internal Data Registers; Local and Recipe. When using local registers, Local Word and Local Bit, the HMI retains the data stored for as long as the HMI is operating. Once power is removed, all data stored inside the HMI is lost.

When using Recipe registers, Recipe Word and Recipe Bit, the HMI retains the data stored even when power is removed. The recipe registers data is maintained using the HMI internal battery.

The following chart shows the memory available in the HMI:

Name	Range	Description
LW (Local Word)	0-9999	These are internal data registers which can be used to store 16-bit, 32-bit, or 64-bit values.
LB (Local Bit)	0-9999	These are internal coil registers which can be used to store two-state values
Ms_LW (Master-Slave Local Word)	0-9999	These are internal data registers used for HMI-HMI Master/Slave communications. When you configure a slave HMI with these addresses, the slave will read/write to the corresponding LW address in the master HMI. This can be used to exchange information between slave and master HMIs.
Ms_LB (Master-Slave Local Bit)	0-9999	These are internal data registers used for HMI-HMI Master/Slave communications. When you configure a slave HMI with these addresses, the slave will read/write to the corresponding LB address in the master HMI. This can be used to exchange information between slave and master HMIs.
RWI (Recipe Word Index)	0-32767	Reserved for recipes. The address referred to by an RWI device is actually the specified address plus an indexed offset. The offset is the value in address LW9000.
RW (Recipe Word)	0-65535	Reserved for recipe registers.
Ms_RW (Master-Slave Recipe Word)	0-65535	Reserved for recipe words when using multiple HMIs
RBI (Recipe Bit Index)	00-2047F	Reserved for recipes. The address referred to by an RBI device is actually the specified address plus an indexed offset. The offset is the value in address LW9000. The bit is addressed as a hex digit placed at the end of the word address.
RB (Recipe Bit)	00-2047F	Reserved for recipe bits. The RB device allows Bit-level access to recipe data. RB devices use same memory area as the RW devices. They reference the first 10 bits of each RW. For example, Bit RBO is Bit 0 in RWO: Bit R10 is Bit 0 in RW1, RB50 is Bit 0 in RW5. The bit is addressed as a hex digit placed at the end of the word address.
Ms_RB (Master-Slave Recipe Bit)	00-4095F	Reserved for recipe bits when using multiple HMIs. The bit is addressed as a hex digit placed at the end of the word address.

Part of the Local Word (LW9000-9999) and Local Bit (LB9000-9999) registers are reserved for internal use by the HMI:

Reserved Local Bit (LB) Table

Address	Function	Access	Description
9000-9009	On after HMI reset	R/W	These bits can be used for any object that needs to be initially set when HMI is initialized.
9010	Recipe Download Indicator	RdOnly	This bit is set when a recipe is downloading to the PLC
9011	Recipe Upload Indicator	RdOnly	This bit is set when a recipe is uploading from a PLC
9012	Recipe Download/Upload Indicator	RdOnly	This bit is set when a recipe is download or uploaded to a PLC
9013	Taskbar Touch indicator	RdOnly	This bit is set when the touch indicator on the taskbar is pressed.
9014	Taskbar CPU indicator	RdOnly	This bit is set when the CPU indicator on the taskbar is pressed.
9015	Taskbar Alarm indicator	RdOnly	This bit is set when the Alarm indicator on the taskbar is pressed.
9016	Printer Error Indicator	RdOnly	This bit is set when the HMI is unable to print to the printer.
9017	Printer Control Bit	R/W	This bit can be used to temporarily disable the printer port on the HMI. Set the bit to disable the printer port. Clear the bit to enable. Note: You must enable the printer option in the Edit…Set System Parameters…General tab menu of EasyBuilder.
9018,9019	Reserved for future use		
9020	Message Board- Pen option	R/W	Set this bit to select Pen mode for the Message Board feature
9021	Message Board- Brush option	R/W	Set this bit to select Brush mode for the Message Board feature
9022	Message Board- Clip option	R/W	Set this bit to select Clip mode for the Message Board feature
9023-9029	Reserved for future use		
9030	Message Board- Pen width = 1	R/W	Set this bit to select Pen Width = 1 pixel for the Message Board feature
9031	Message Board- Pen width = 2	R/W	Set this bit to select Pen Width = 2 pixels for the Message Board feature
9032	Message Board- Pen width = 3	R/W	Set this bit to select Pen Width = 3 pixels for the Message Board feature
9033-9039	Reserved for future use		
9040	Hide Fast SelectionWindow	R/W	Set this bit to hide the Fast Selection window; clear it to display the Fast Selection window
9041	Hide TaskBar	R/W	Set this bit to hide the TaskBar window; clear it to display the TaskBar window
9042	Hide TaskBar Buttons	R/W	Set this bit to hide the TaskBar buttons; clear it to display the TaskBar buttons
9043	Hide All (Fast Select,TaskBar, and TaskBarButtons)	R/W	Set this bit to hide the Fast Selection window, the TaskBar window, and the TaskBar buttons; clear it to display the three objects
9044	Restore system parameters from recipe words	R/W	Set this bit ON will restore these system parameters from Reserved Recipe Word. After a restore, the system will clear this bit off.
9045	Reset HMI	Write only	Set this bit ON to reset (warm boot) the HMI. After a reset, the HMI reboots and clears this bit.
9046	Low Security Level	R/W	Security Error: Latches to a 1 when there is an attempt to enter a higher security level page when the security level is not set. User must clear it.
9047-9049	Reserved for future use		
9050	Toshiba Protocol: T© control bit	R/W	For Toshiba protocol driver. When this bit is on, write T© as ON Or OFF T1/T1S user's manual. NOTE: In case of Timer/Counter register write, the Timer/Counter's device data (2 bytes each) corresponding to the Timer/Counter's register should be added. If the Timer/Counter's device is set to ON, it should be '01'. Otherwise, it should be 00.
9051	TouchScreen Control during disabled backlight	R/W	Set this bit to disable the touch screen when the backlight is off.

9052	PLC Control Object; Change Window option – disable clearing PLC register	R/W	Set this bit to prevent the HMI from clearing the PLC register used for the Change Window option
9053,9054	Reserved for future use		
9055	Loss of communications to PLC; Writing to PLC register control	R/W	If HMI loses connection to the PLC while it is trying to write to a PLC register, set this bit to continue to send Write command after communications is reestablished. Clearing this bit prevents the HMI from issuing the Write command after connection is reestablished. 0: Any write to PLC command will be killed. 1: Any write to PLC command will be continuously retired.
9056	Loss of communications to PLC; Touchscreen control	R/W	If HMI loses connection to the PLC, setting this bit disables the touchscreen until communications is reestablished.
9057-9059	Reserved for future use		
9060	Keypad Control; AI or NI located on left side of keypad window	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the left side of the keypad. The HMI clears this bit when user presses the Enter or ESC keys. Note: this bit can be used to control Keypad popup.
9061	Keypad Control; AI or NI located on left side of keypad window	RdOnly	Same as LB9060
9062	Keypad Control; AI or NI located on left top side of keypad window	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the left side of the keypad on the top half. The HMI clears this bit when user presses the Enter or ESC keys.
9063	Keypad Control; AI or NI located on left bottom side of keypad window	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the left side of the keypad on the bottom half. The HMI clears this bit when user presses the Enter or ESC keys.
9064	Keypad Control; AI or NI located on right side of keypad window	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the right side of the keypad. The HMI clears this bit when user presses the Enter or ESC keys.
9065	Keypad Control; AI or NI located on right side of keypad window	RdOnly	Same as LB9064
9066	Keypad Control; AI or NI located on right top side of keypad window	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the right side of the keypad on the top half. The HMI clears this bit when user presses the Enter or ESC keys.
9067	Keypad Control; AI or NI located on right bottom side of keypad window	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the right side of the keypad on the bottom half. The HMI clears this bit when user presses the Enter or ESC keys.
9068	Keypad Control; AI or NI located on any side of keypad window	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on any side of the keypad. The HMI clears this bit when user presses the Enter or ESC keys.
9069	Keypad Control; AI or NI located on any side of keypad window	RdOnly	Same as LB9068
9070-9079	Reserved for future use		
9080	Keypad control; AI or NI located on TOP side of keypad window.	RdOnly	The OIT sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the TOP side of the touchscreen. The OIT clears this bit when user presses the Enter or ESC keys.
9081	Keypad Control: AI or NI located on BOTTOM side of keypad window.	RdOnly	The HMI sets this bit when user edits a Numeric Input Object or ASCII Input Object located on the BOTTOM side of the touchscreen. The HMI clears this bit when user presses the Enter or ESC keys.
9082-9089	Reserved for future use		
9090	Clear Data Event Log	R/W	2.6.0 -- This bit is used to clear the event data. Set to clear data. After it is set, the HMI automatically clears the bit after data is erased.
9091	Contrast Up	Write	V 2.6.0 -- Use to increment the contrast.
9092	Contrast Down	Write	V. 2.6.0 -- Use to decrement the contrast.
9093-9099	Reserved for future use		

9100-9227	PLC Communication Status	R/W	V 2.6.0 -- These bits are mapped to PLC Nodes 0-127. Bit changes to 1 when the communication times out. Write 0 to resume the communication.
9228-9355	AUX device Communication Status	R/W	V 2.6.0 -- These bits are mapped to AUX device nodes 0-127. Bit changes to 1 when the communication times out. Write to 0 to resume communication.
9356-9959	Reserved for future use		
9360	Compact Flash Status	RdOnly	0:CF not detected 1:CF detected
9361	Recipe Transfer from Compact Flash	R/W	Set to 1 to start transfer.
9362-9999	Reserved for future use		

Reserved Local Word (LW) Table

Address	Function	Access	Description
9000	Recipe Index Base	R/W	RWI and RBI use this index to access recipe data. Maximum value for correct functionality is 32,767.
9001	Reserved for future use		
9002,9003	Maximum Value – Numeric Input Object	RdOnly	This represents the maximum value allowed when a Numeric Input Object is active. If no Numeric Input Object is active, the value is 0.
9004,9005	Minimum Value – Numeric Input Object	RdOnly	This represents the minimum value allowed when a Numeric Input Object is active. If no Numeric Input Object is active, the value is 0.
9006	Message Board Operation Mode	RdOnly	This value represents the current operation mode of the Message Board. 0=open, 1=brush, 2=clipping
9007	Message Board Pen Width	RdOnly	This value represents the current pent width in pixels of the Message Board
9008	Message Board Pen Color	RdOnly	This value represents the current pen color (0-255) of the Message Board
9009	Reserved for future use		
9010	Local second, (bcd, 0-59)	R/W	When the local Word Option is selected for RTC source in the Edit System Parameters-General settings dialog box of EasyBuilder, this is the location of the seconds parameter. Use the Data Transfer Object to copy the PLC RTC seconds value to this register.
9011	Local minute, (bcd, 0-59)	R/W	When the local Word Option is selected for RTC source in the Edit System Parameters-General settings dialog box of EasyBuilder, this is the location of the minutes parameter. Use the Data Transfer Object to copy the PLC RTC minutes value to this register.
9012	Local hour, (bcd, 0-23)	R/W	When the local Word Option is selected for RTC source in the Edit System Parameters-General settings dialog box of EasyBuilder, this is the location of the hours parameter. Use the Data Transfer Object to copy the PLC RTC hours value to this register.
9013	Local date, (bcd, 0-31)	R/W	When the local Word Option is selected for RTC source in the Edit System Parameters-General settings dialog box of EasyBuilder, this is the location of the date parameter. Use the Data Transfer Object to copy the PLC RTC date value to this register.
9014	Local month, (bcd, 0-11)	R/W	When the local Word Option is selected for RTC source in the Edit System Parameters-General settings dialog box of EasyBuilder, this is the location of the month parameter. Use the Data Transfer Object to copy the PLC RTC month value to this register.

9015	Local year, (bcd, 0-9999)	R/W	When the local Word Option is selected for RTC source in the Edit System Parameters-General settings dialog box of EasyBuilder, this is the location of the year parameter. Use the Data Transfer Object to copy the PLC RTC year value to this register.
9016	Local day, (bcd, 0-6)	R/W	When the local Word Option is selected for RTC source in the Edit System Parameters-General settings dialog box of EasyBuilder, this is the location of the day parameter. Use the Data Transfer Object to copy the PLC RTC year value to this register.
9017-9019	Reserved for future use		
9020	Object queue, Item Number	RdOnly	The object queue represents the number of objects on the HMI screen which requires updating. If this number exceeds 1000, then this may slow PLC communication throughout. If this happens, we recommend that you change the design of the window displayed.
9021-9033	Reserved for future use		
9034-9035	System Time (unit as 0.1 seconds)	R/W	Represents a continuous revolving timer with 0.1 seconds time base.
9036-9039	Reserved for future use		
9040,9041	Window Security Level password	Write Only	This 32-bit register is used to enter passwords to change the window security level. Must use binary decimal 32-bit format. Write only access because HMI changes value back to 0 after password is read.
9042	Window Security Level	RdOnly	This is the current level of window security (0, 1, or 2).
9043	Force Window Security Level	R/W	This register can be used to 'force' the window security level to a lower level without entering a password.
9044	Momentary Key Release; Popup Window conflict	R/W	This register affects how a Set Bit Object configured as Momentary behaves if a popup window is displayed over the key as it is being released. If LW9044 = 0: the coil assigned to the momentary key is cleared when the operator releases the key, even if a popup window is now overlaid on top of the key. If LW9044 = 1: in this mode, no popup windows can be displayed until after the momentary key is released. Any requests by the HMI or the PLC to display a popup window are ignored. If LW9044 = 2: the momentary key coil remains set if a popup window covers the momentary key before it is released.
9045-9049	Reserved for future use		
9050	Base Window ID	RdOnly	This is the number of the base window currently displayed.
9051	Base Window- Reserved	RdOnly	This register is used by the PLC control object if the master register 9050 is used to mimic the master current window in the Slave unit. Remember, a windows PLC control object writes back in the next address.
9052-9053	Reserved for future use		
9054	Report Printout Option	R/W	0: print text, meters, and trends 1: print text, meters, trends, and shapes but without pattern 2: print text, meters, trends, and bitmaps 3: print text, meters, trends, bitmaps, and shapes but without pattern. 4: print everything
9055	PLC Control Object word offset	R/W	This controls how the PLC Control Object using the Change Window or Report Printout features is used. If a non-zero number is placed into this register, then the HMI will automatically add this number to the number retrieved from the PLC register assigned to the PLC Control Object to arrive at the new screen to display or printout. For example, if LW9055 = 10 and the PLC Control Object/Change Window feature reads the value of 2 from the PLC register assigned to it, then the HMI will place Screen #12 onto the display, (LW9055+2 = 12). Note: when writing to the next consecutive register, the HMI will still only write the value placed into the PLC register assigned to the PLC Control object.
9056	Reserved for future use		

9057	Event Log Database Item Size	RdOnly	Management information: The size of every item.
9058-9059	Event Log Database Size	RdOnly	Management information: The size of the database, the size includes management information. (Total_Item*Item_Size+management_info_size).
9060-9075	Numeric Input Object and ASCII Input Object data	RdOnly	These 16 registers represent the latest entries, when the HMI operator enters a numeric or ASCII value into one of these objects. Register 9075 is the latest entry.
9076-9079	Reserved for future use		
9080-9085	Project name	RdOnly	Twelve bytes are used to store the project name. Use the ASCII Data Object to display on HMI.
9086-9087	Project size	RdOnly	Four bytes are used to store the size of the current project (in bytes). Use the Numeric Data Object (Decimal Display) to display on HMI
9088-9089	Project size (Kbytes)	RdOnly	Four bytes are used to store the size of the current project (in bytes). Use the Numeric Data Object (Decimal Display) to display on HMI.
9090-9091	Compiler Version ID	RdOnly	Four bytes are used to store the compiler version of the current project. Use the Numeric Data Object (Decimal Display) to display on HMI.
9092	Project Compiled Date: Year	RdOnly	This register contains the year date when the project was last compiled. Use the Numeric Data Object (Decimal Display) to display on HMI.
9093	Project Compiled Date: Month	RdOnly	This register contains the month date when the project was last compiled. Use the Numeric Data Object (Decimal Display) to display on HMI.
9094	Project Compiled Date: Day	RdOnly	This register contains the day of month date when the project was last compiled. Use the Numeric Data Object (Decimal Display) to display on HMI.
9095-9099	Reserved for future use		
9100	Indirect Window Object; window #	RdOnly	This register stores the window number requested using the Indirect Window Object
9101	Indirect Window Object; Offset	R/W	This register can be used to store an offset that is added to LW9100 when using the Indirect Window Object
9102-9129	Reserved for future use.		
9130	Change language	R/W	V 2.5.0 -- This register changes the language set (Values:0-3)
9135	Battery Voltage (BIN)	RdOnly	V 2.6.0 - Li-Battery Measurement (If value = 1175 = 3.02V) (If value <1070 or 2.75V, then replace the battery) Create a Numeric Display for LW:9135, set "Input high=1175" and set "engineering high = 302" Then display should show proper voltage.
9136	CF Recipe Transfer	RdOnly	0:inactive 1:Transfer in process 2:Transfer complete 3:Transfer failed
9137-9999	Reserved for future use		

Reserved Recipe Word (RW) Table

Address	Function	Access	Description
60000	Real Time Clock (RTC); Seconds	R/W	This register is used to store the seconds count for the optional RTC; BCD code, valid values 0-59
60001	Real Time Clock (RTC); Minutes	R/W	This register is used to store the minutes count for the optional RTC; BCD code, valid values 0-59
60002	Real Time Clock (RTC); Hour	R/W	This register is used to store the hour count for the optional RTC; BCD code, valid values 0-23
60003	Real Time Clock (RTC); Day Of Month	R/W	This register is used to store the day of the month for the optional RTC; BCD code, valid values 0-31

60004	Real Time Clock (RTC); Month	R/W	This register is used to store the month for the optional RTC; BCD code, valid values 0-11
60005	Real Time Clock (RTC); Year	R/W	This register is used to store the year for the optional RTC; BCD code, valid values 0-9999
60006	Real Time Clock (RTC); Day of the Week	R/W	This register is used to store the day of the week for the optional RTC; BCD code, valid values 0-6 (with 0=Sunday)

Non-volatile Storage of System Parameters

System Parameters can be stored in the non-volatile memory held in the Recipe module. To enable this feature, see the Hardware tab of the Set System Parameters dialog. Under the Edit menu, select System Parameters.

The HMI must be restarted in order to activate any changes.

The System Parameters are divided into three groups: PLC Settings, General Settings, Security Settings.

PLC Settings

Address	Function	Description
60040	Serial port	0:RS232, 1:RS485
60041	Baud Rate	0:9600, 1:192002:38400, 3:576004:115200
60042	Data Bits	0:7, 1:8
60043	Parity	0:none, 1:even, 2:odd
60044	Stop Bits	0:1, 1:2
60045	HMI Station Address	0-255
60046	PLC Station Address	0-255
60047	Multiple HMI	0:none, 1:master, 2:slave
60048	HMI-HMI link speed	0:38400, 1:115200
60049	PLC time out constant	
60050	PLC block pack	

General Settings

Address	Function	Access	Description
60051	Task button: Attribute		
60052	Task button: Position		
60053	Task button: background color		
60054	Task button: Text		
60055	Alarm bar: Pixel per scroll		
60056	Alarm bar: Scroll speed		
60057	Number of windows		
60058	Reserved		
60060	Startup window No.		
60061	Backlight saver	R/W	This register is used to change the timer on the backlight saver mode. (0=disable - Backlight is on all the time; 1-255=minutes before the backlight goes off.)
60062	Cursor color		
60064	Buzzer Enable	R/W	This register can change the state of the internal buzzer. (0=disable buzzer from sounding; 1=enable buzzer to sound when screen is touched).
60065	Common window: popup window		

60066	Common window: Attribute		
60067	Extra No. of Events		
60068	RTC Source		
60069	Printer:Printer		
60070	Message board window No.		

Security Settings

Address	Function	Access	Description
60071	Security Control	R/W	0=Disables all security control passwords 1=Enables Security control of windows
60072	Password for: Level 0	R/W	Contains password value (2 words, 32-bits)
60074	Password for: Level 1	R/W	Contains password value (2 words, 32-bits)
60076	Password for: Level 1	R/W	Contains password value (2 words, 32-bits)

If Recipe: Save System Parameter: is set to "Yes", then set a bit of LB:9044 to ON will restore these system parameters from these recipe word memory.

Representing PLC Coil Registers

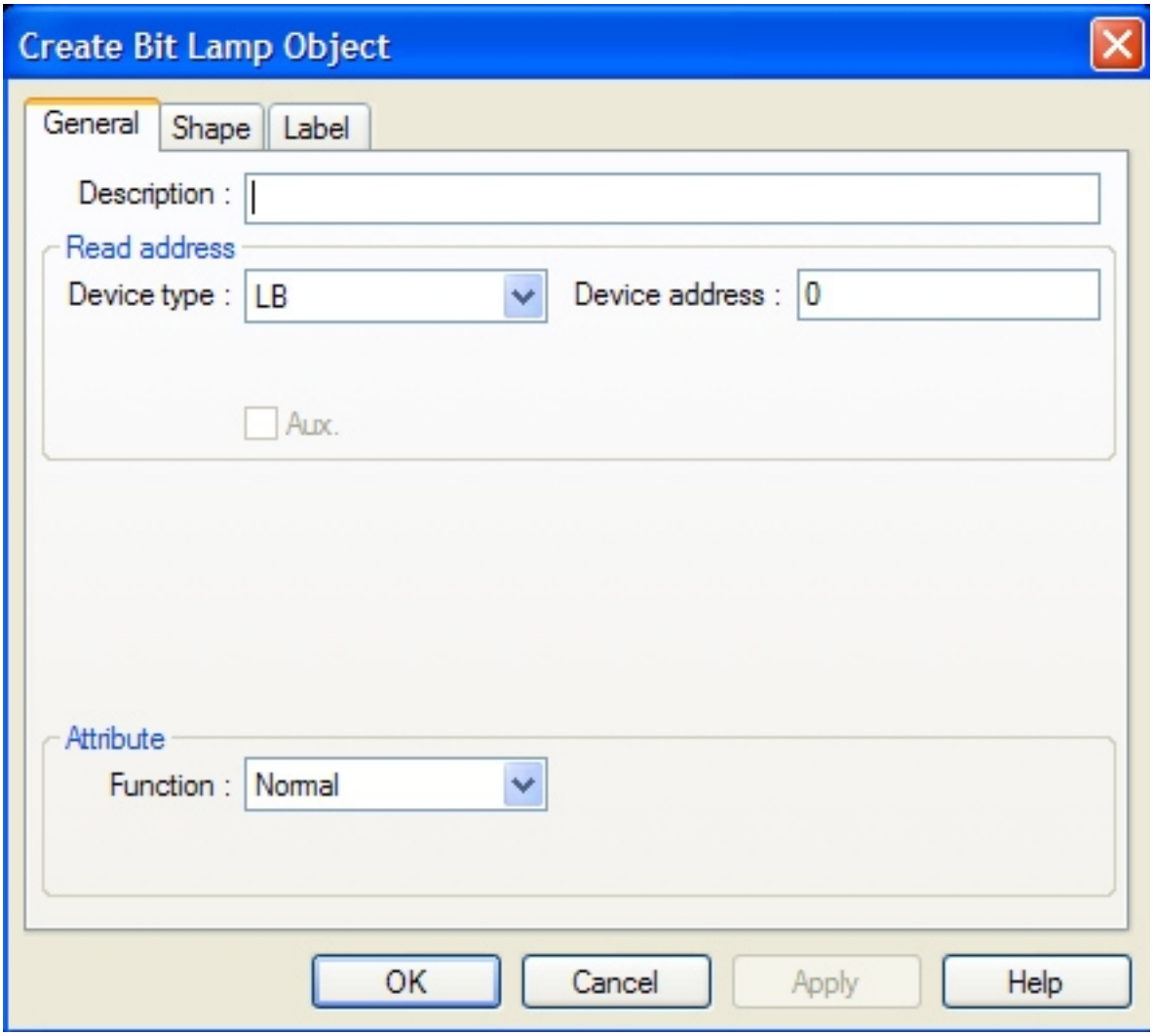
PLC coils (or binary registers) and internal coils of the HMI are represented using three parts: the Bit Lamp Object, the Set Bit Object, and the Toggle Switch Object. Each has a particular function that makes them unique but they are all constructed and used in essentially the same manner.

The Bit Lamp Object

The Bit Lamp Object is used to represent the value of a PLC coil. The object continuously reads the PLC coil and displays the corresponding shape or bitmap that is tagged to the On or Off state of the coil.

To create a Bit Lamp Object 

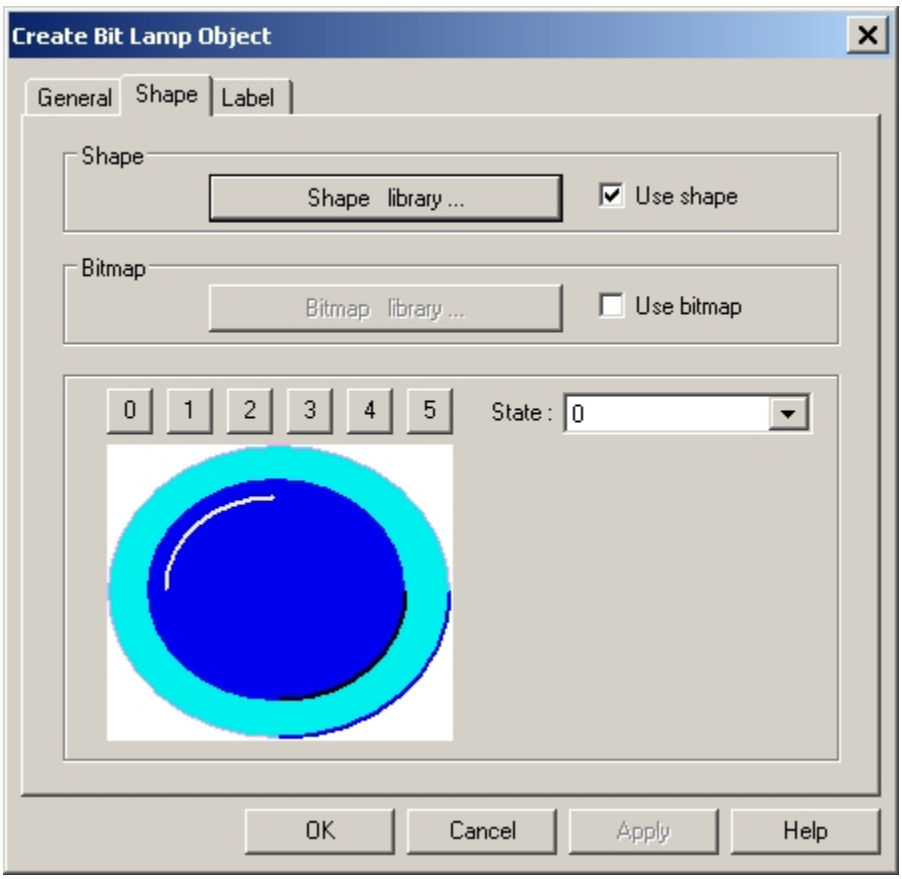
1. From the **Parts** menu, click **Bit Lamp**. Or click the **Bit Lamp** icon in the Part1 toolbar. The Create Bit Lamp Object dialog box appears.



2. Use the **Description:** box to enter a title for the Bit Lamp part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC coil or HMI internal memory address.
4. In the **Attribute** frame, select how you want the Bit Lamp object to operate:

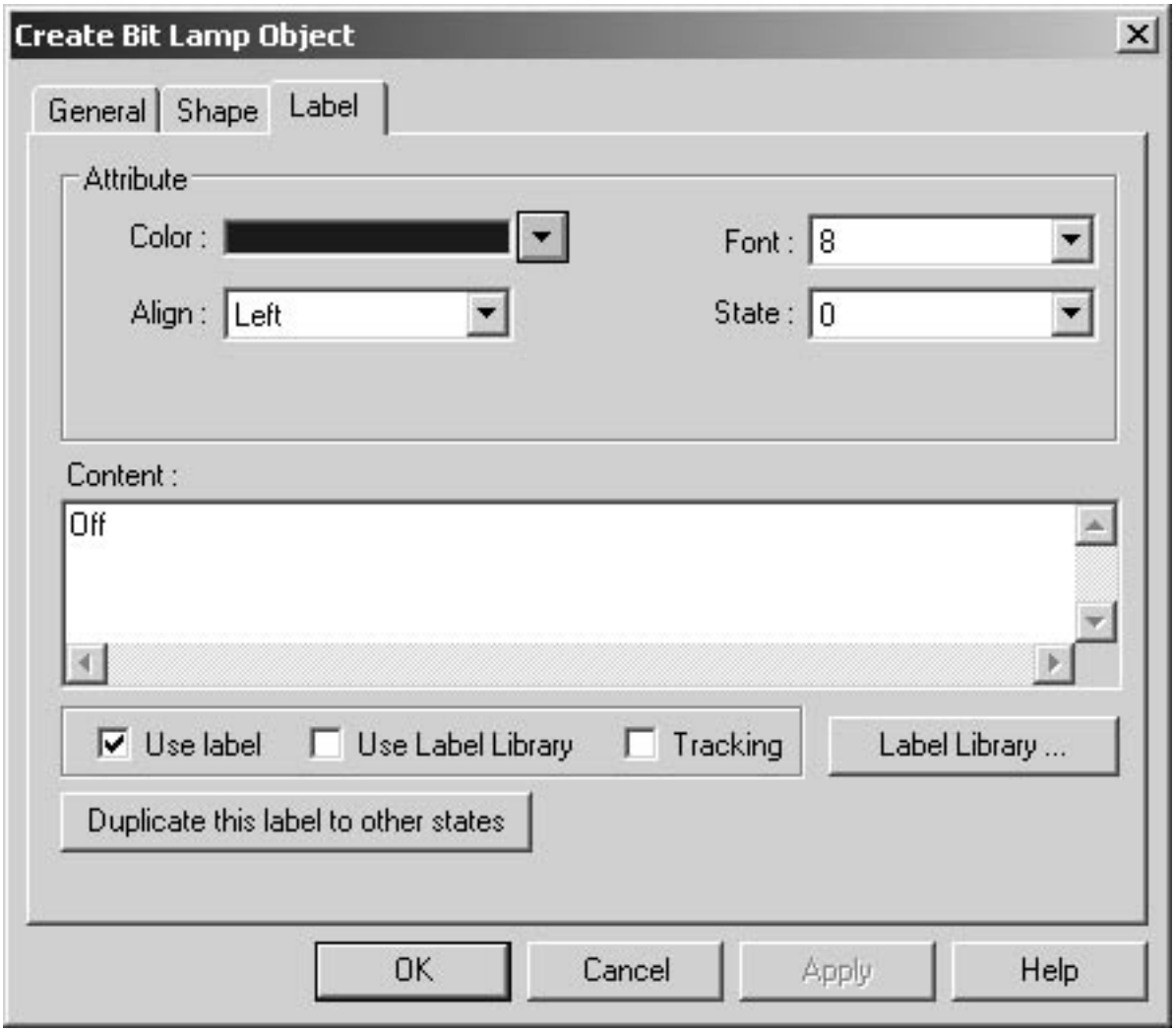
Function	Option	Description
Normal		Displays State 0 object when coil is clear; displays State 1 object when coil is set.
Blinking on State 1		Displays State 0 object when coil is clear; blinks State 1 object when coil is set.
Blinking on State 0		Displays State 0 object when coil is clear, blinks State 0 object when coil is set.
	Break time:	This is the time period for the blinking option in 1/10ths of a second increments. 0 = default of 1/10th second. Range: 0 to 2,147,483,647

5. Click the **Shape** tab to display the Shape form



6. Select either a shape or bitmap to represent the Bit Lamp object. If you need more information on how to do this, consult Chapter 6 “Creating Graphics Objects”.

7. Click the **Label** tab to display the Label form.



8. Click the **Use label** checkbox if you want to use a label. Click the **Duplicate this label to other states button** if you want to use the same label for all states.
9. You can have a different label for each state of the Bit Lamp. In the **State:** box, enter the state that you wish to edit, then enter the label for that state in the **Content:** box. Do the same for other states.
10. Enter the color that you want for the label in the **Color:** box. You can select a different color for each label state.
11. Enter how you want the label to be positioned in the **Align:** box. You can select a different position for each label state.
12. Enter the size of the label in the **Font:** box. You can select a different font for each label state.
13. Click the **Tracking** checkbox if you want the labels for all states to follow or 'track' with the movement of one label when it is moved after the Bit Lamp object is placed onto the window.
14. Click **OK**. The Create Bit Lamp Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location of the window.
15. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Now click on the label again. Now only the label is highlighted, allowing you to move it without moving the

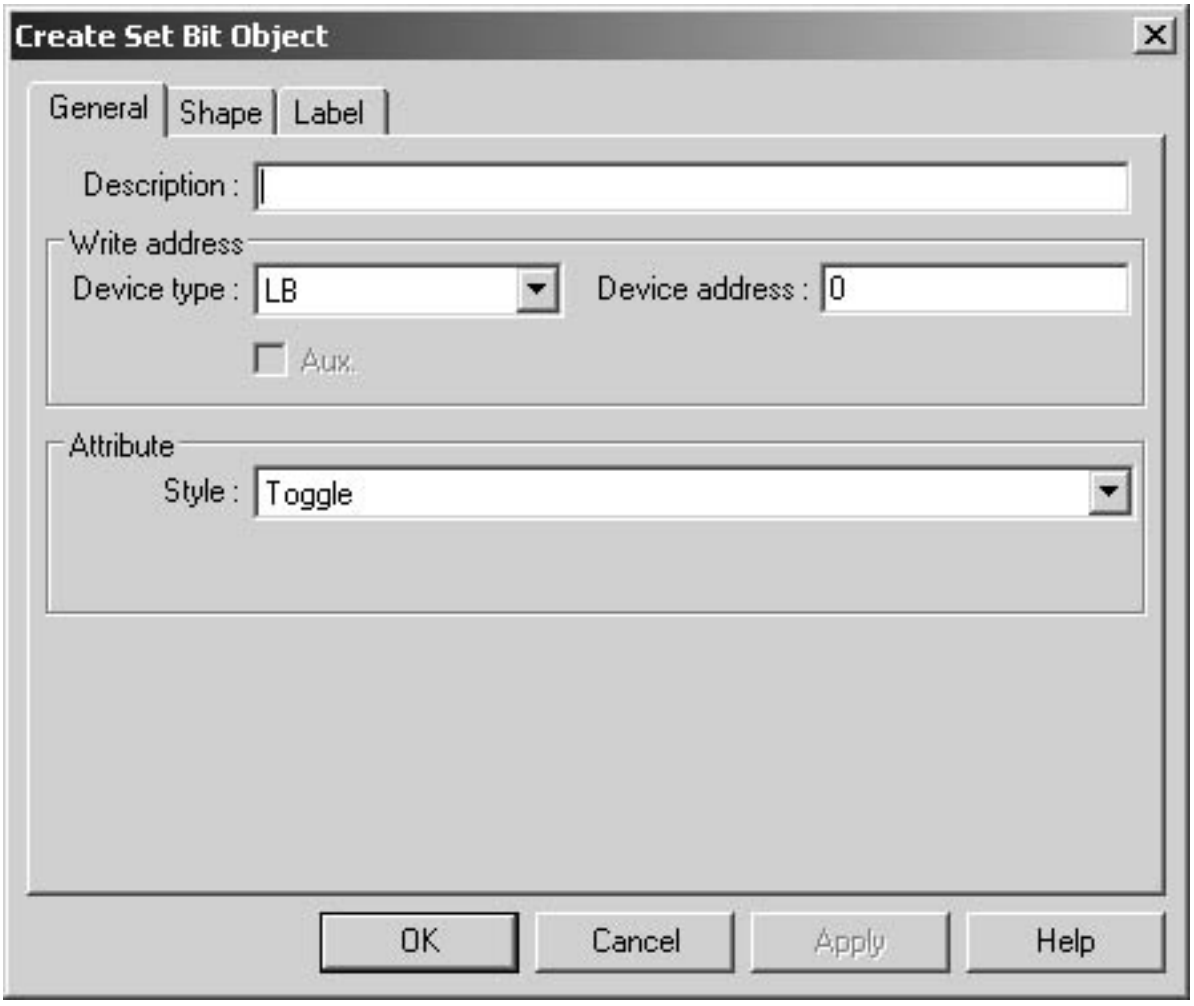
part. Note: if you double-click (click twice rapidly) then you will not highlight the label but rather enter the Bit Lamp Object's Attribute dialog box.

The Set Bit Object

The Set Bit Object is used to write a value to a PLC coil. The touchscreen object displays the State 0 shape until pressed. When pressed, it displays the State 1 shape and executes the function that was assigned to it.

To create a Set Bit Object 

1. From the **Parts** menu, click **Set Bit**. Or click the **Set Bit** icon in the Part1 toolbar. The Create Set Bit Object dialog box appears.



2. Use the **Description:** box to enter a title for the Set Bit part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Write address** frame, select the PLC coil or HMI internal memory address.
4. In the **Attribute** frame, select how you want the Set Bit object to operate:

Style	Option	Description
ON		Sets a PLC coil when pressed.
OFF		Clears a PLC coil when pressed.
Toggle		Alternates between setting and clearing a PLC coil.
Momentary		Sets a PLC coil when pressed; clears a PLC coil when released
Set ON at window open		Automatically sets a PLC coil when the window that the part is located in is opened.
Set OFF at window open		Automatically clears a PLC coil when the window that the part is located in is opened.
Periodical toggle		Automatically sets and clears a PLC coil as long as the window that the part is located in is open (or minimized).
	Interval:	Used for the Periodical toggle feature to determine the period at which the bit is toggled. Range: 0.0 to 3.0 seconds
Set ON at window close		Automatically sets a PLC coil when the window that the part is located in is closed.
Set OFF at window close		Automatically clears a PLC coil when the window that the part is located in is closed.
Set ON at window maximized		Automatically sets a PLC coil when the window that the part is located in is maximized.
Set OFF at window maximized		Automatically clears a PLC coil when the window that the part is located in is maximized.
Set ON at window minimized		Automatically sets a PLC coil when the window that the part is located in is minimized.
Set OFF at window minimized		Automatically clears a PLC coil when the window that the part is located in is minimized.
Set ON at backlight off		Automatically sets a PLC coil when the backlight screen saver feature causes the HMI backlight to shut off.
Set OFF at backlight off		Automatically clears a PLC coil when the backlight screen saver feature causes the HMI backlight to shut off.
Set ON at Enter success		Sets the bit when the numeric keypad's Enter key has been pressed, and the data is valid. The Set Bit object must be placed on the same window as the Numeric Input object.
Set OFF at Enter success		Clears the bit when the numeric keypad's Enter key has been pressed, and the data is valid. the Set Bit object must be placed on the same window as the Numeric Input object.

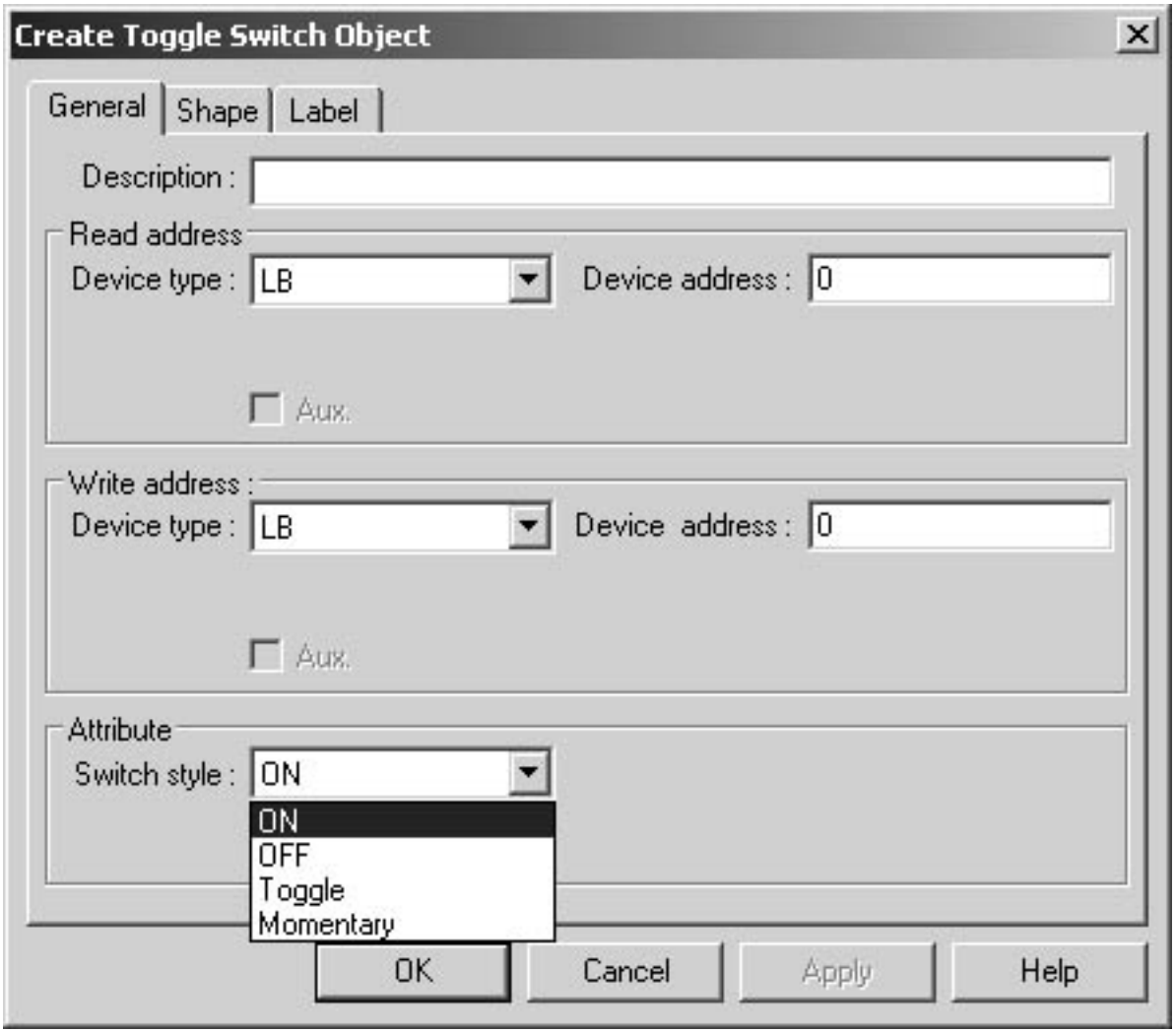
5. Refer to the Bit Lamp Object section for information on using the **Shape** tab. Note that it is not required to use a shape or a bitmap. In some modes such as **Set ON at window open** or **Periodical toggle** you may prefer to place the part on the window without a shape or bitmap tied to it.
6. Refer to the Bit Lamp Object section for information on using the **Label** tab.
7. Click **OK**. The Create Set Bit Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the location on the window that you want it.
8. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Now click on the label again. Now only the label is highlighted, allowing you to move it without moving the part. Note: if you double-click (click twice rapidly) then you will not highlight the label but rather enter the Set Bit Object's Attribute dialog box.

The Toggle Switch Object

The Toggle Switch Object is used to represent the value of one PLC coil while able to write a value to the same or another PLC coil. The object continuously reads one PLC coil and displays the corresponding shape or bitmap that is tagged to the On or Off state of the coil. Also the touchscreen object when pressed, executes the function that was assigned to it.

To create a Toggle Switch Object 

1. From the **Parts** menu, click **Toggle Switch**. Or click the **Toggle Switch** icon in the Part1 toolbar. The Create Toggle Switch Object dialog box appears.



2. Use the **Description:** box to enter a title for the Toggle Switch part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC coil or HMI internal memory address that the part continuously reads.
4. In the **Write address** frame, select the PLC coil or HMI internal memory address that the part writes to when pressed.
5. In the **Attribute** frame, select how you want the Toggle Switch object to operate:

Switch Style	Description
ON	Sets a PLC coil when pressed.
OFF	Clears a PLC coil when pressed.
Toggle	Alternates between setting and clearing a PLC coil.
Momentary	Sets a PLC coil when pressed; clears a PLC coil when released

6. Refer to the Bit Lamp Object section for information on using the **Shape** tab

7. Refer to the Bit Lamp Object section for information on using the **Label** tab.
8. Click **OK**. The Create Toggle Switch Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location in the window.
9. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Now click on the label again. Now only the label is highlighted, allowing you to move it without moving the part. Note: if you double-click (click twice rapidly) then you will not highlight the label but rather enter the Toggle Switch Object's Attribute dialog box.

Representing PLC Data Registers

PLC registers and internal data registers of the HMI are represented using nine parts:

- Word Lamp Object
- Set Word Object
- Multistate Switch Object
- Numeric Data Object
- Numeric Input Extend Object
- ASCII Data Object
- ASCII Input Extend Object
- Moving Shape Object
- Animation Object

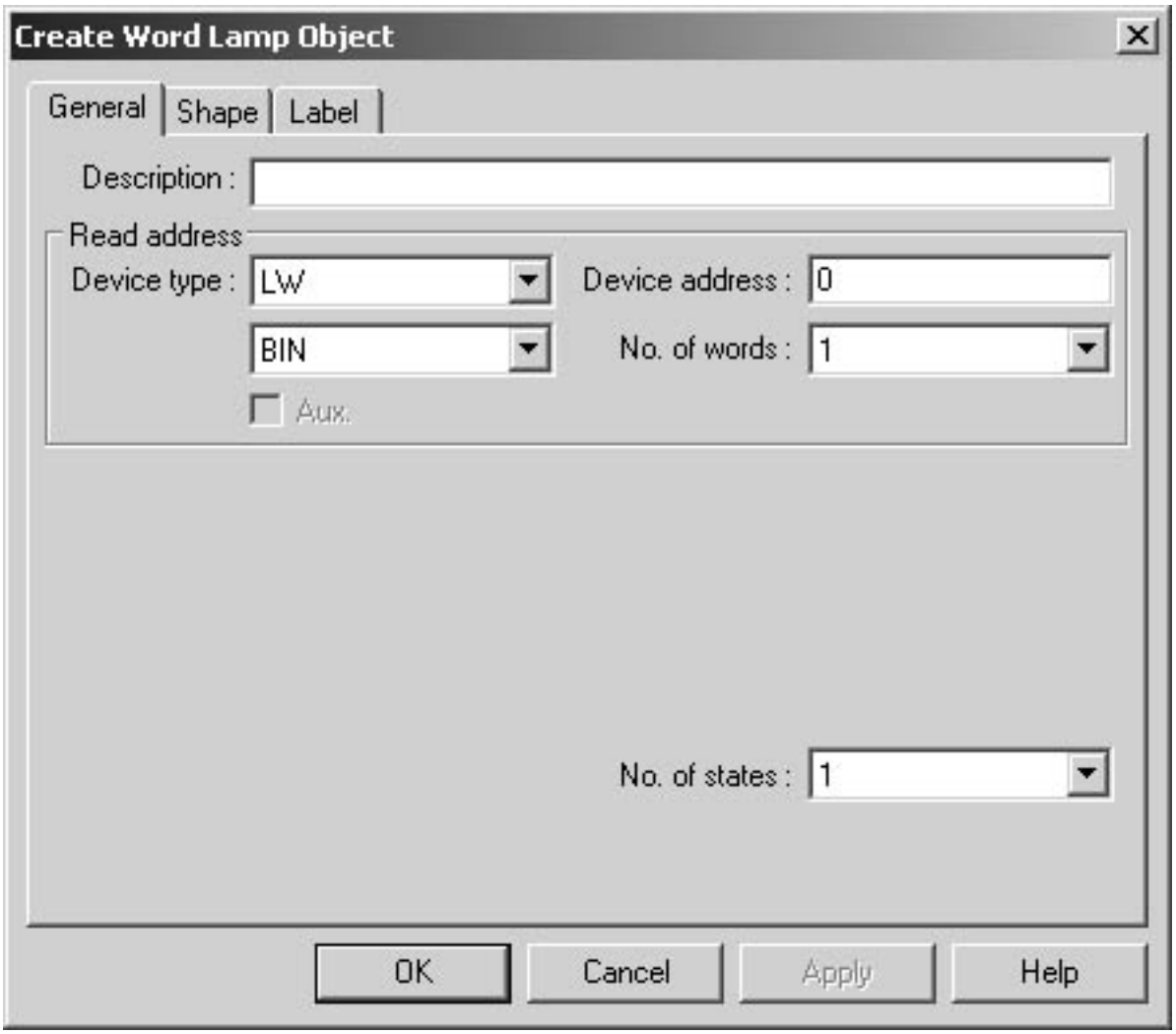
They are constructed in much the same manner as the parts described in the previous section. With these parts, however, you can represent data from 16-bit, 32-bit, or even 64-bit data registers.

The Word Lamp Object

The Word Lamp Object is used to represent the value of a PLC register. The object continuously reads the PLC register and displays the corresponding shape or bitmap that is tagged to the number of states defined.

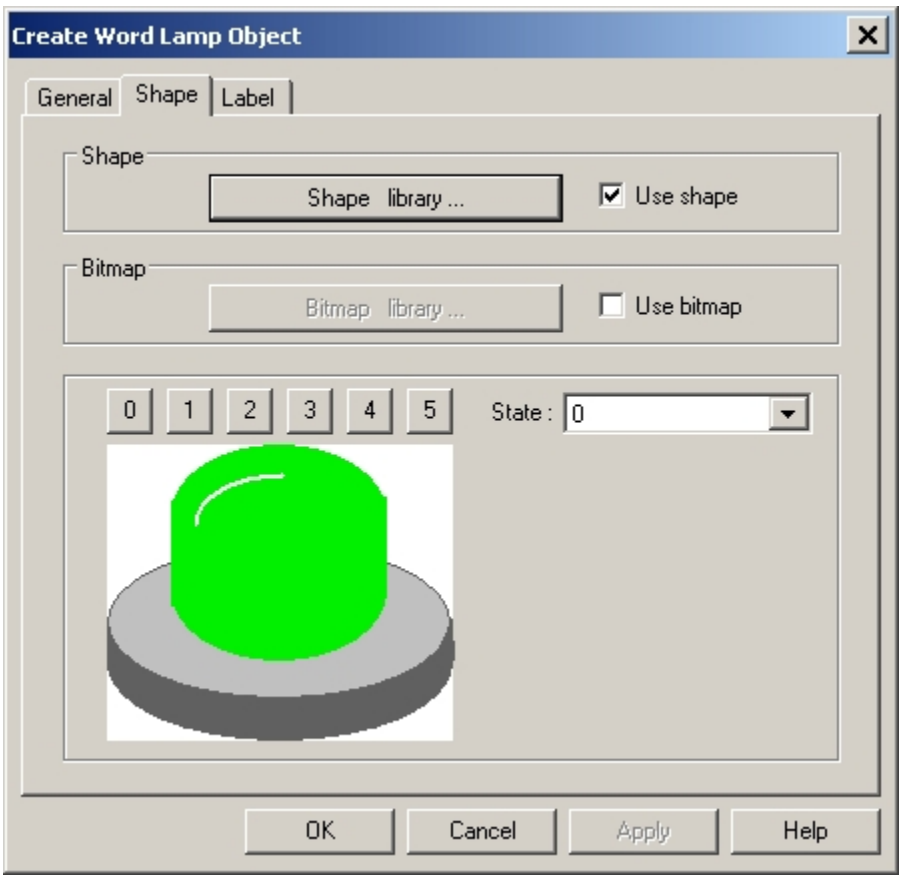
To create a Word Lamp Object 

1. From the **Parts** menu, click **Word Lamp**. Or click the **Word Lamp** icon in the Part1 toolbar. The Create Word Lamp Object dialog box appears.



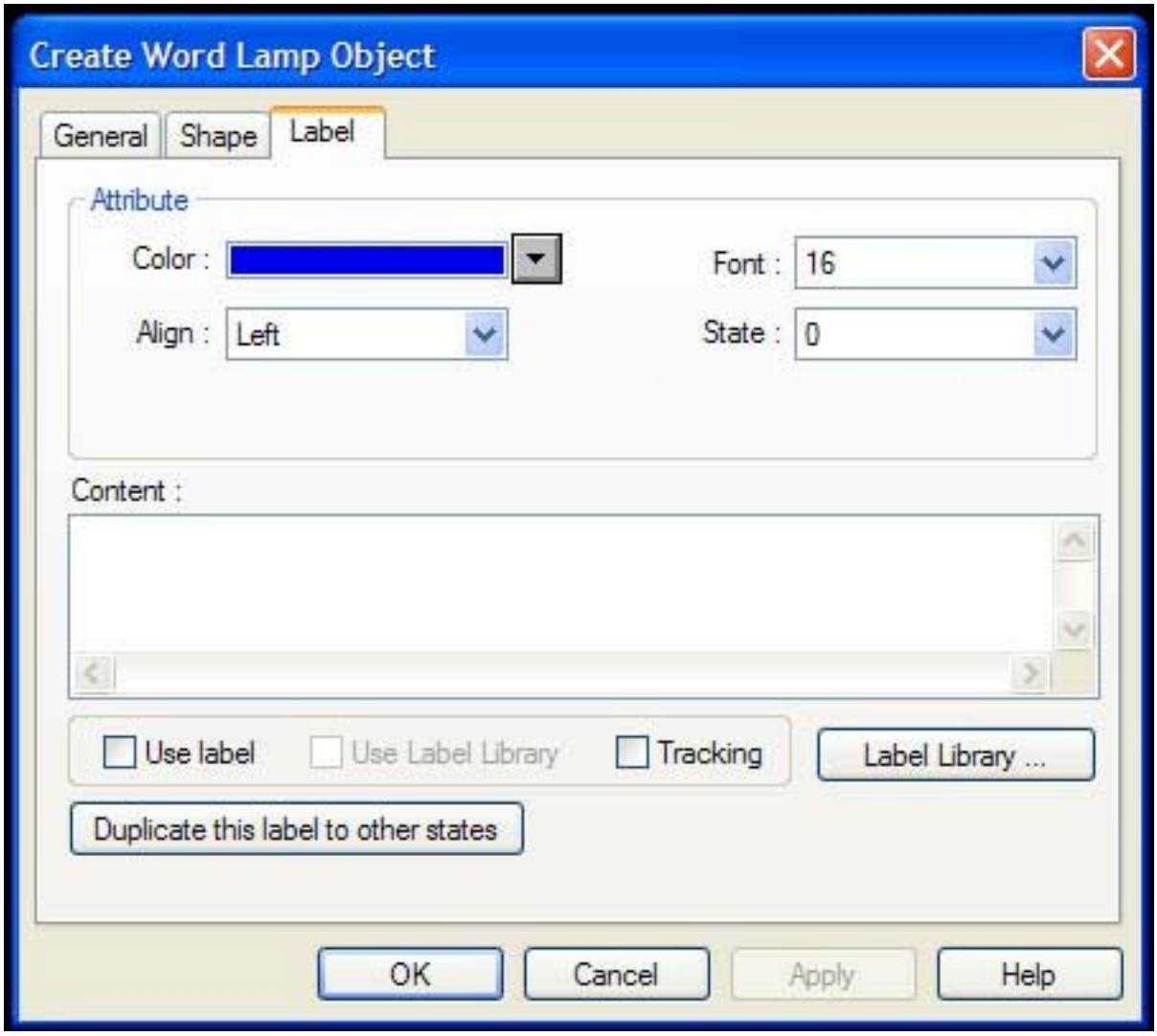
2. Use the **Description:** box to enter a title for the Word Lamp part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC register or HMI internal memory address. You can select **BIN** (binary) or **BCD** format. In the **No. of words:** box, select 1 for monitoring a 16-bit register.
4. In the **No. of states:** box, select from 1 to 32 states. If the actual value in the PLC register is out-of-range for the number of states specified, then no shape or bitmap is displayed.

5. Click the **Shape** tab to display the Shape form.



6. Select either a shape or bitmap to represent the Word Lamp object. If you need more information on how to do this, consult Chapter 6 “Creating Graphics Objects”.

7. Click the **Label** tab to display the Label form.



8. Click the **Use Label** checkbox if you want to use a label. Click the **Duplicate this label to other states** button if you want to use the same label for all states.
9. You can have a different label for each state of the Word Lamp. In the **State:** box, enter the state that you wish to edit, then enter the label for that state in the **Content:** box. Do the same for the other states.
10. Enter the color that you want for the label in the **Color:** box. You can select a different color for each label state.
11. Enter how you want the label to be positioned in the **Align:** box. You can select a different position for each label state.
12. Finally, enter the size of the label in the **Font:** box. You can select a different font for each label state.
13. Click the **Tracking** checkbox if you want the labels for all states to follow or 'track' with the movement of one label when it is moved after the Word Lamp object is placed onto the window.
14. Click **OK**. The Create Word Lamp Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.

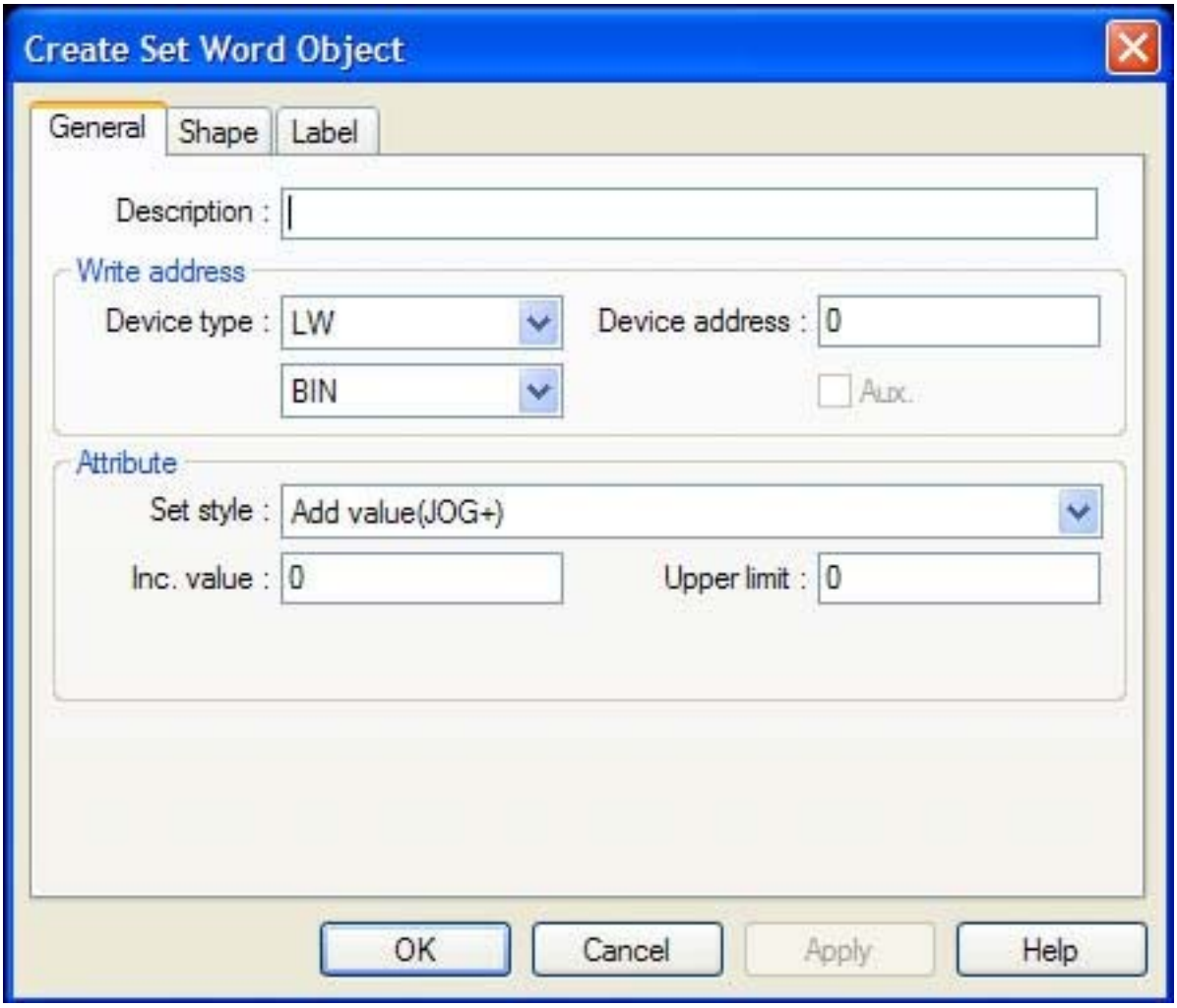
15. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Now click on the label again. Now only the label is highlighted, allowing you to move it without moving the part. Note: if you double-click (click twice rapidly) then you will not highlight the label but rather enter the Word Lamp Object's Attribute dialog box.

The Set Word Object

The Set Word Object is used to write a value to a PLC register. The touchscreen object displays the State 0 shape until pressed. When pressed, it displays the State 1 shape and executes the function that was assigned to it.

▶ **To create a Set Word Object** 

1. From the **Parts** menu, click **Set Word**. Or click the **Set Word** icon in the Part1 toolbar. The Create Set Word Object dialog box appears.



2. Use the **Description:** box to enter a title for the Set Word part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Write address** frame, select the PLC coil or HMI internal memory address. You can select **BIN** (binary) or **BCD** format.

4. In the **Attribute** frame, select how you want the Set Word object to operate:

Set Style	Option	Description
Set Constant		Places a 16-bit number in a PLC register when pressed.
	Set Value:	Constant used. Range: -32768 to 65535
Add value (JOG+)		Increases the value in a 16-bit PLC register by a step value each time it is pressed.
	Inc.value:	Amount of the step value. Range: 1 to 32767
	Upper limit:	The upper range allowed. Range: -32768 to 32767
Sub value (JOG-)		Increases the value in a 16-bit PLC register by a step value each time it is pressed. If the key is held down, then it will continue to increase by the step value at a predefined rate.
	Inc. value:	Amount of the step value. Range: 1 to 32767
	Upper limit:	The upper range allowed. Range: -32768 to 32767
	JOG delay:	The amount of delay after the key is held down before automatic incrementing occurs. Range: 0.1 to 1.5 seconds
	JOG speed:	The rate at which the automatic incrementing occurs. Range: 0.1 to 1.5 seconds
JOG—		Decreases the value in a 16-bit PLC register by a step value each time it is pressed. If the key is held down, then it will continue to decrease by the step value at a predefined rate.
	Dec. value:	Amount of the step value. Range: 1 to 32767
	Bottom limit:	The lowest range allowed. Range: -32768 to 32767
	JOG delay:	The amount of delay after the key is held down before automatic decrementing occurs. Range: 0.1 to 1.5 seconds
	JOG speed:	The rate at which the automatic decrementing occurs. Range: 0.1 to 1.5 seconds
Set at window open		Automatically places a constant value into a PLC register when the window that the part is located in is opened.
	Set value:	Constant used. Range: -32768 to 65535
Set at window close		Automatically places a constant value into a PLC register when the window that the part is located in is closed.
	Set value:	Constant used. Range: -32768 to 65535
Periodical JOG++		Continuously increases the value in a 16-bit PLC register by a step value at a predefined rate as long as the window that the part is located on is open. This is very similar to a self resetting timer.
	Inc. value:	Amount of the step value. Range: 1 to 32767
	Upper limit:	The upper range allowed. Range: -32768 to 32767
	Break time:	The rate at which the automatic incrementing occurs. Range: 0.1 to 25.5 seconds in 0.1 second increments
Periodical JOG—		Continuously decreases the value in a 16-bit PLC register by a step value at a predefined rate as long as the window that the part is located on is open. This is very similar to a self resetting timer.
	Dec. value:	Amount of the step value. Range: 1 to 32767
	Bottom limit:	The lowest range allowed. Range: -32768 to 32767
	Break time:	The rate at which the automatic decrementing occurs. Range: 0.1 to 25.5 seconds in 0.1 second increments
Periodical bounce		Automatically increases the value in a 16-bit PLC register by a step value at a predefined rate as long as the window that the part is located on is open. When the upper limit is reached, the value automatically decreases until it reaches 0, then repeats the process continuously.
	Inc. value:	Amount of the step value. Range: 1 to 32767
	Upper limit:	The upper range allowed. Range: 1 to 32767
	Break time:	The rate at which the automatic incrementing occurs. Range: 0.1 to 25.5 seconds in 0.1 second increments

Step up		Continuously increases the value in a 16-bit PLC register by 1 from a specified low limit at a predefined rate as long as the window that the part is located on is open. When the high limit is reached, the process repeats starting at the low limit again.
	Low limit:	The lowest range allowed. Range: -32768 to 65535
	Upper limit:	The upper range allowed. Range: -32768 to 65535
	Break time:	The rate at which the automatic incrementing occurs. Range: 0.1 to 25.5 seconds in 0.1 second increments
Step down		Continuously decreases the value in a 16-bit PLC register by 1 from a specified high limit at a predefined rate as long as the window that the part is located on is open. When the low limit is reached, the process repeats starting at the high limit again.
	Low limit:	The lowest range allowed. Range: -32768 to 65535
	Upper limit:	The upper range allowed. Range: -32768 to 65535
	Break time:	The rate at which the automatic decrementing occurs. Range: 0.1 to 25.5 seconds in 0.1 second increments
Set at window maximized		Automatically places a 16-bit constant value into a PLC register when the window that the part is located in is maximized.
	Set value:	Constant used. Range: -32768 to 65535
Set at window minimized		Automatically places a 16-bit constant value into a PLC register when the window that the part is located in is minimized.
	Set value:	Constant used. Range: -32768 to 65535
Set at backlight off		Automatically places a 16-bit constant into a PLC register when the backlight screen saver feature causes the HMI backlight to shut off.
	Set value:	Constant used. Range: -32768 to 65535
Set at Enter success		Writes the specified value to the Write Address when the numeric keypad's Enter key is pressed, and the data is valid. The Set Word object must be placed on the same window as the Numeric Input Object.
Set at Enter fail		Writes the specified value to the write address when the numeric keypad's Enter key is pressed, and the data is invalid. The set word object must be placed on the same window as the numeric input object.

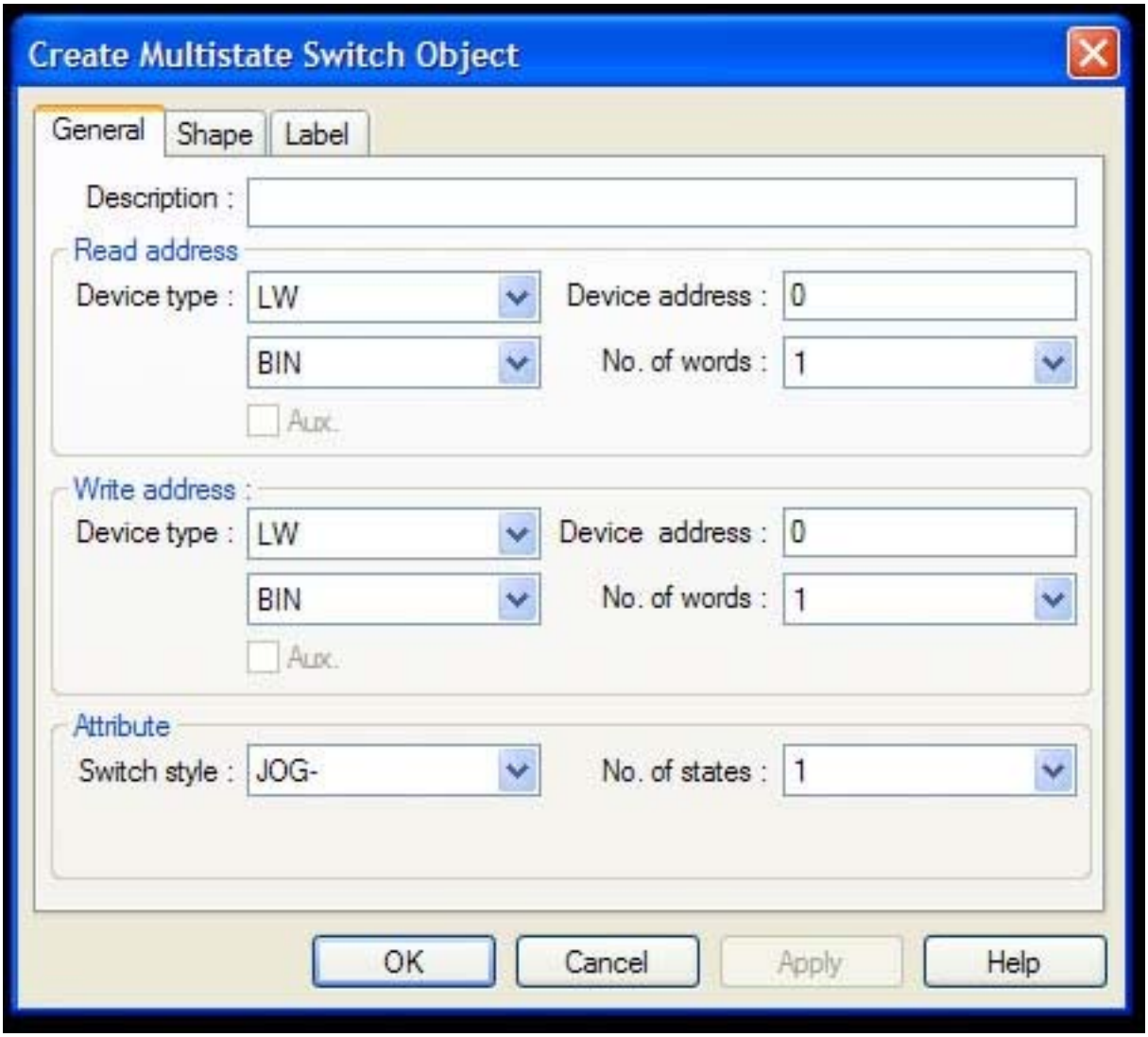
5. Refer to the Bit Lamp Object section for information on using the **Shape** tab. Note that it is not required to use a shape or a bitmap. In some modes such as **Set at window open** or **Periodical JOG++** you may prefer to place the part on the window without a shape or bitmap tied to it.
6. Refer to the Bit Lamp Object section for information on using the **Label** tab.
7. Click **OK**. The Create Set Word Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
8. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Click on the label again. Now only the label is highlighted, allowing you to move it without moving the part.
   Note: if you double-click (click twice rapidly) then you will not highlight the label but rather

The MultiState Switch Object

The MultiState Switch Object is used to represent the value of one 16-bit PLC register while writing a 16-bit value to the same or another PLC register each time the object is pressed. The object continuously reads one PLC register and displays the corresponding shape or bitmap that is tagged to the defined states of the register. The touchscreen object when pressed, executes the function that was assigned to it.

To create a MultiState Switch Object

1. From the **Parts** menu, click **MultiState Switch**. Or click the **MultiState Switch** icon in the Part1 toolbar. The Create MultiState Switch Object dialog box appears.



2. Use the **Description:** box to enter a title for the MultiState Switch part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the 16-bit PLC register or HMI internal memory address that the part continuously reads. You can select **BIN** (binary) or **BCD** format.
4. In the **Write address** frame, select the 16-bit PLC register or HMI internal memory address that the part writes to when pressed. You can select **BIN** (binary) or **BCD** format.
5. In the **Attribute** frame, select how you want the MultiState Switch object to operate:

Switch Style	Option	Description
JOG+		Increments the PLC register by 1 each time the key is pressed. When the highest state is reached, the key resets to State 0.
	No. of states:	Range: 1 to 32
JOG-		Decrements the PLC register by 1 each time the key is pressed. When the State 0 is reached, the key resets to the highest state.
	No. of states:	Range: 1 to 32

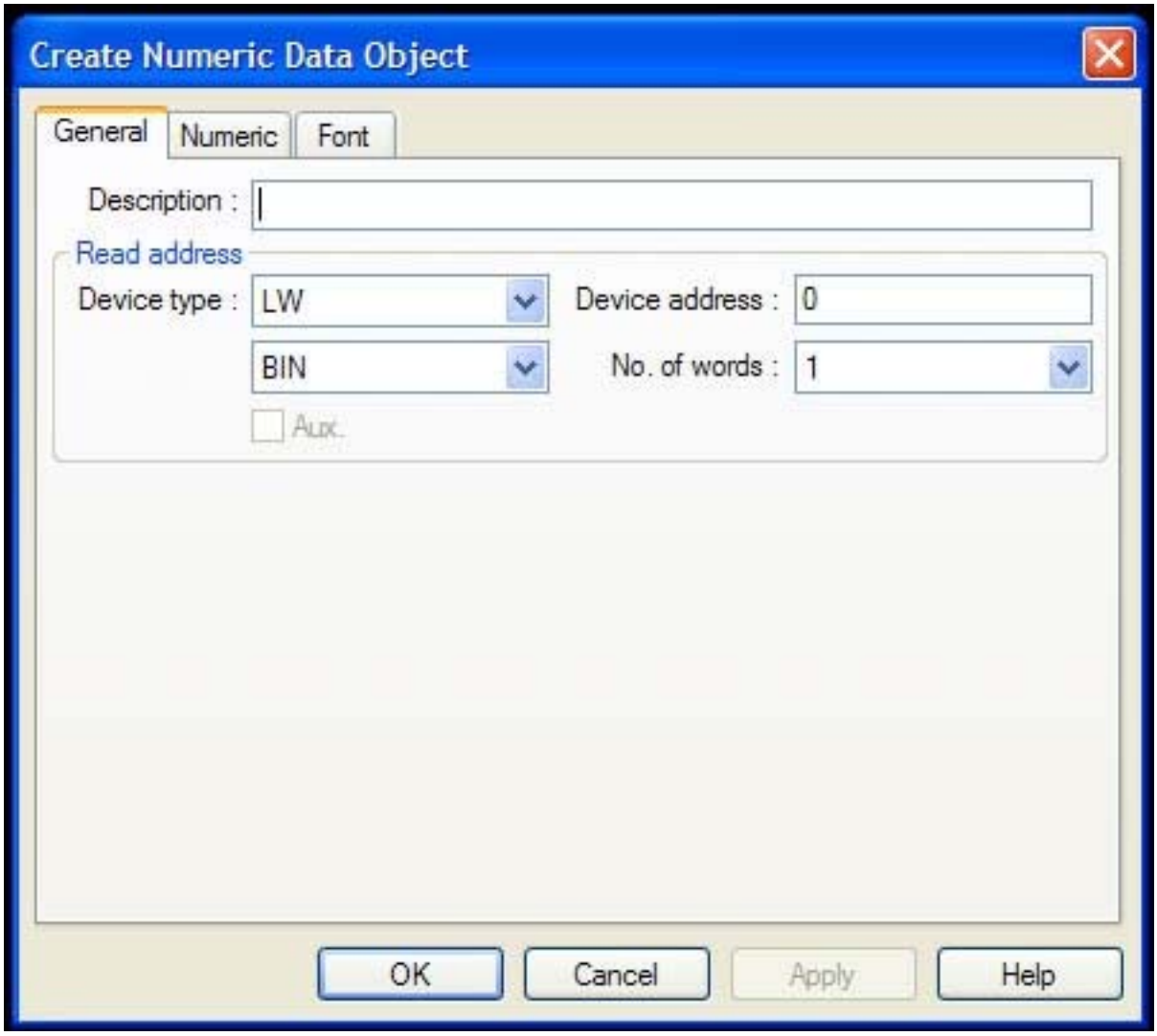
6. Refer to the Bit Lamp Object section for information on using the **Shape** tab.
7. Refer to the Bit Lamp Object section for information on using the **Label** tab.
8. Click **OK**. The Create MultiState Switch Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
9. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Now click on the label again. Now only the label is highlighted, allowing you to move it without moving the part. Note: if you double-click (click twice rapidly) then you will not highlight the label but rather enter the MultiState Switch Object's Attribute dialog box.

The Numeric Data Object

The Numeric Data Object is used to display the numeric value of a PLC register. The object continuously reads the PLC register and displays the corresponding numeric data in the format specified. The PLC register can be a 16-bit, 32-bit, or a 64-bit word.

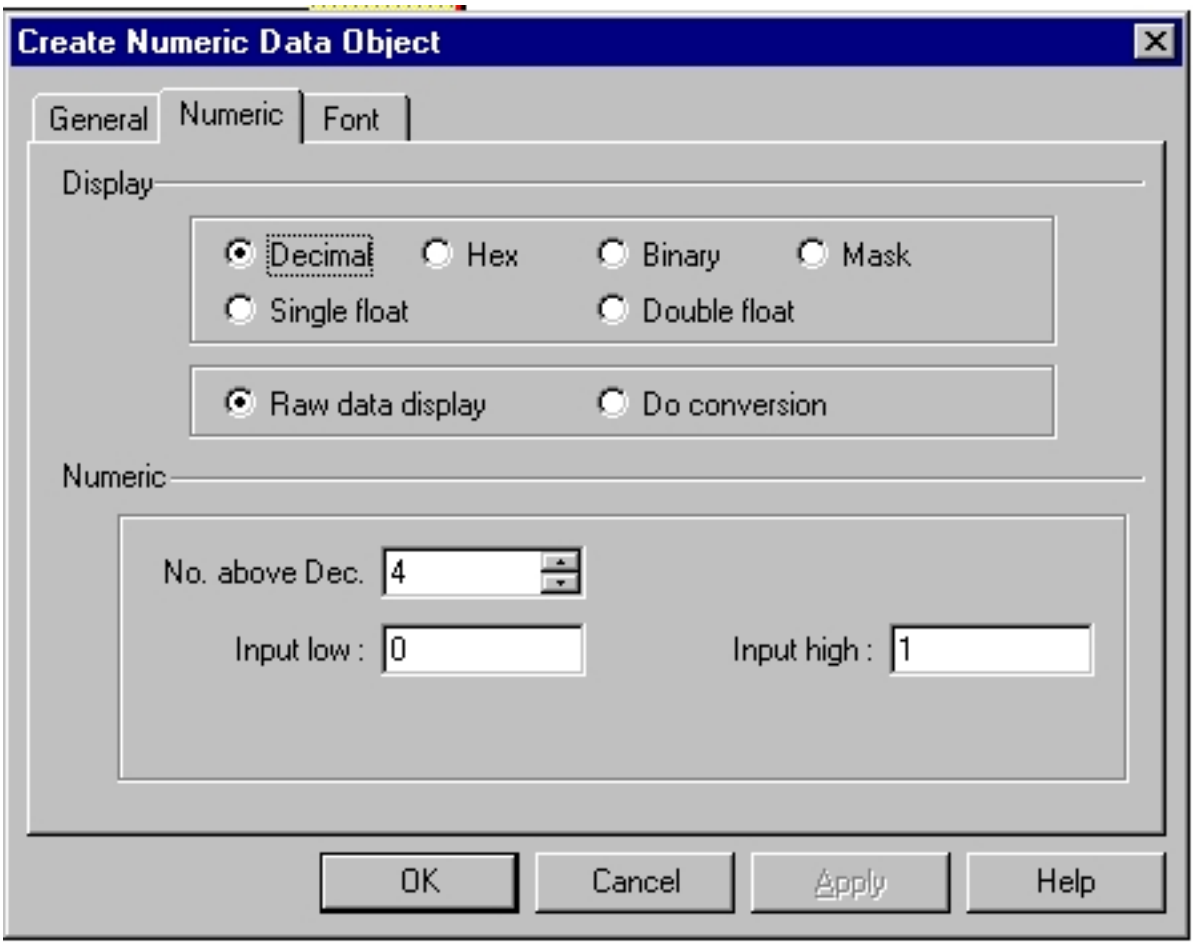
To create a Numeric Data Object

1. From the **Parts** menu, click **Numeric Data**. Or click the **Numeric Data** icon in the Part1 toolbar. The Create Numeric Data Object dialog box appears.



2. Use the **Description:** box to enter a title for the Numeric Data part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC register or HMI internal memory address that the part continuously reads. You can select **BIN** (binary) or **BCD** format. Select **No. of words:** that the part should read, (1= 16 bits, 2= 32 bits, 4= 64 bits).

4. Click the **Numeric** tab to display the Numeric form.



5. In the **Display** attribute box, select the format type you wish to use. The options are:

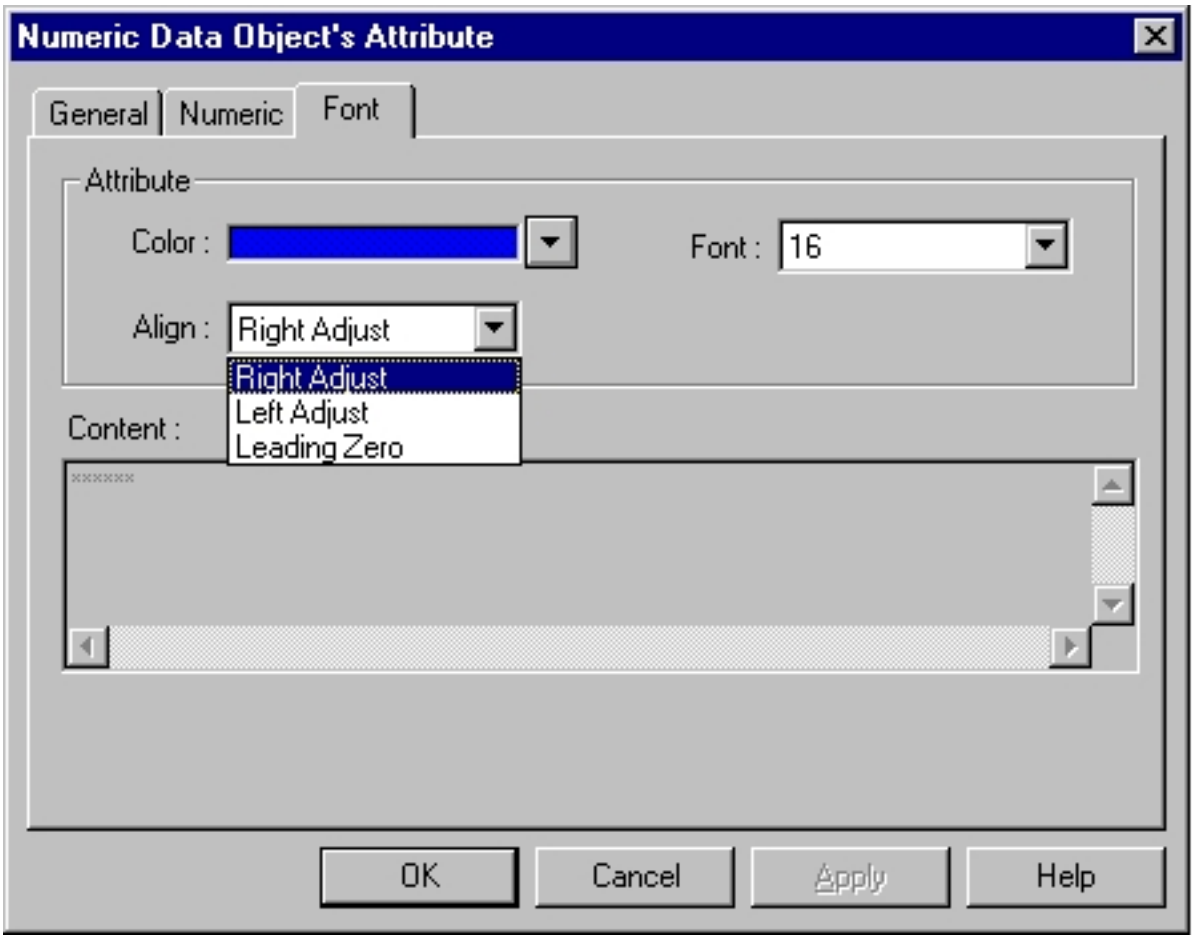
Format	Range	No. of words	Description
Decimal	-32768 to +32767	1	signed 16 bit format
	-999999999 to +2147483647	2	signed 32 bit format
Hex	0000 to FFFF	1	hexadecimal 16 bit format
	00000000 to FFFFFFFF	2	hexadecimal 32 bit format
Binary	00… to 11… (16 bits)	1	binary 16 bit format
	00… to 11… (32 bits)	2	binary 32 bit format
Mask	*****	1,2	masks the value with asterisks
Single Float	-999999999.9999999999 to 9999999999.9999999999	2	EEE 754 format; single precision
Double Float	-999999999.9999999999 to 9999999999.9999999999	4	IEEE 754 format; double precision

6. If **Decimal** format is selected, then you have the option of selecting **Raw data display** or **Do conversion**. Use **Raw data display** when you wish to display the true numeric value of the register. Select **Do conversion** when you wish to scale the value.

7. In the **Numeric** attribute box, select how you want the data to be represented. The **Input low:** and **Input high:** boxes don't apply unless Decimal format with Do conversion is selected. The options are:

Format	Options	Range	Description
Decimal (with Raw data display)	No. above Dec.:	1-10	specifies number of digits to display (for example, 2 digits displays -9 to 99); any number that cannot be displayed with the specified number of digits is represented with asterisks.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.
Decimal (with Do conversion)	No. above Dec.:	1-10	specifies number of digits to the left of the decimal point.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.
	Input low:	-2147483648 to +2147483647	This is the lowest raw data value that is in the PLC data register
	Input high:	-2147483648 to +2147483647	This is the highest raw data value that is in the PLC data register
	Engineering low:	-2147483648 to +2147483647	This is the lowest data value that you would like displayed on the HMI.
	Engineering high:	-2147483648 to +2147483647	This is the highest data value that you would like displayed on the HMI.
Hex	No. above Dec.:	Not Apply (NA)	4 digits are used for 16 bit; 8 digits for 32 bit.
Binary	No. above Dec.:	NA	16 digits are used for 16 bit; 32 digits for 32 bit.
Mask	No. above Dec.:	1-10	the number of asterisks to display
Single Float	No. above Dec.:	1-10	specifies number of digits to the left of the decimal point.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.
Double Float	No. above Dec.:	1-10	specifies number of digits to the left of the decimal point.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.

8. Click the **Font** tab to display the Font form.



9. Enter the color that you want for the digits displayed in the **Color:** box
10. Enter how you want the digits to be positioned in the **Align:** box. The choices are:

- **Right Adjust** - aligns the number to the right margin of the part.
- **Left Adjust** - aligns the number to the left margin of the part.
- **Leading Zero** - inserts leading zeroes

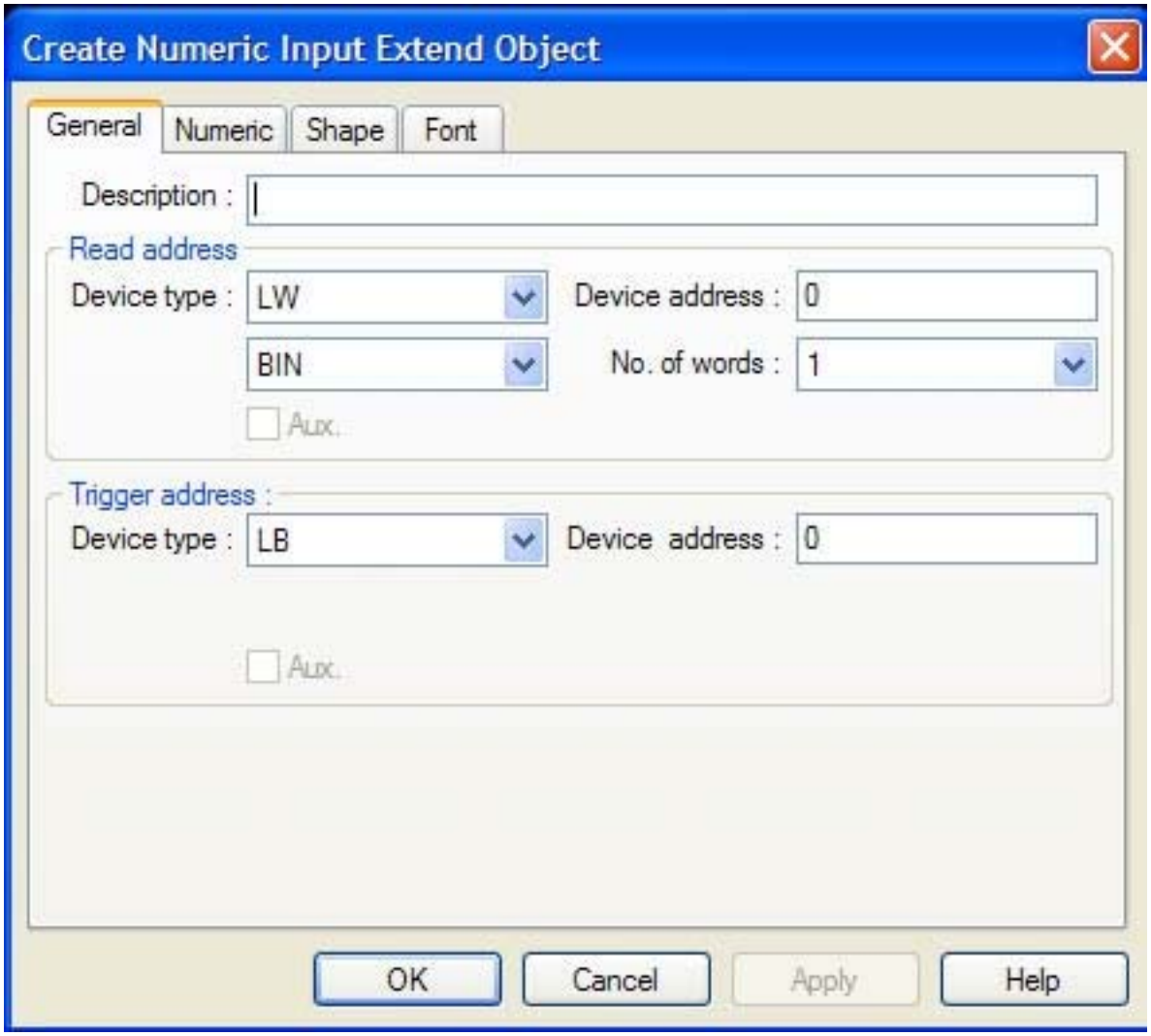
11. Finally, enter the size of the digits in the **Font:** box.
12. Click **OK**. The Numeric Data Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
13. Once the part is placed onto the window, you can modify the attributes by double-clicking on the part to display the Numeric Data Object's Attribute dialog box.

The Numeric Input Extend Object

The Numeric Input Extend Object is used to read and write numeric values to a PLC register. The object continuously reads the PLC register and displays the corresponding value in the format specified. The PLC register can be a 16-bit, 32-bit, or a 64-bit word. By using the keypad, the HMI operator can enter a new value.

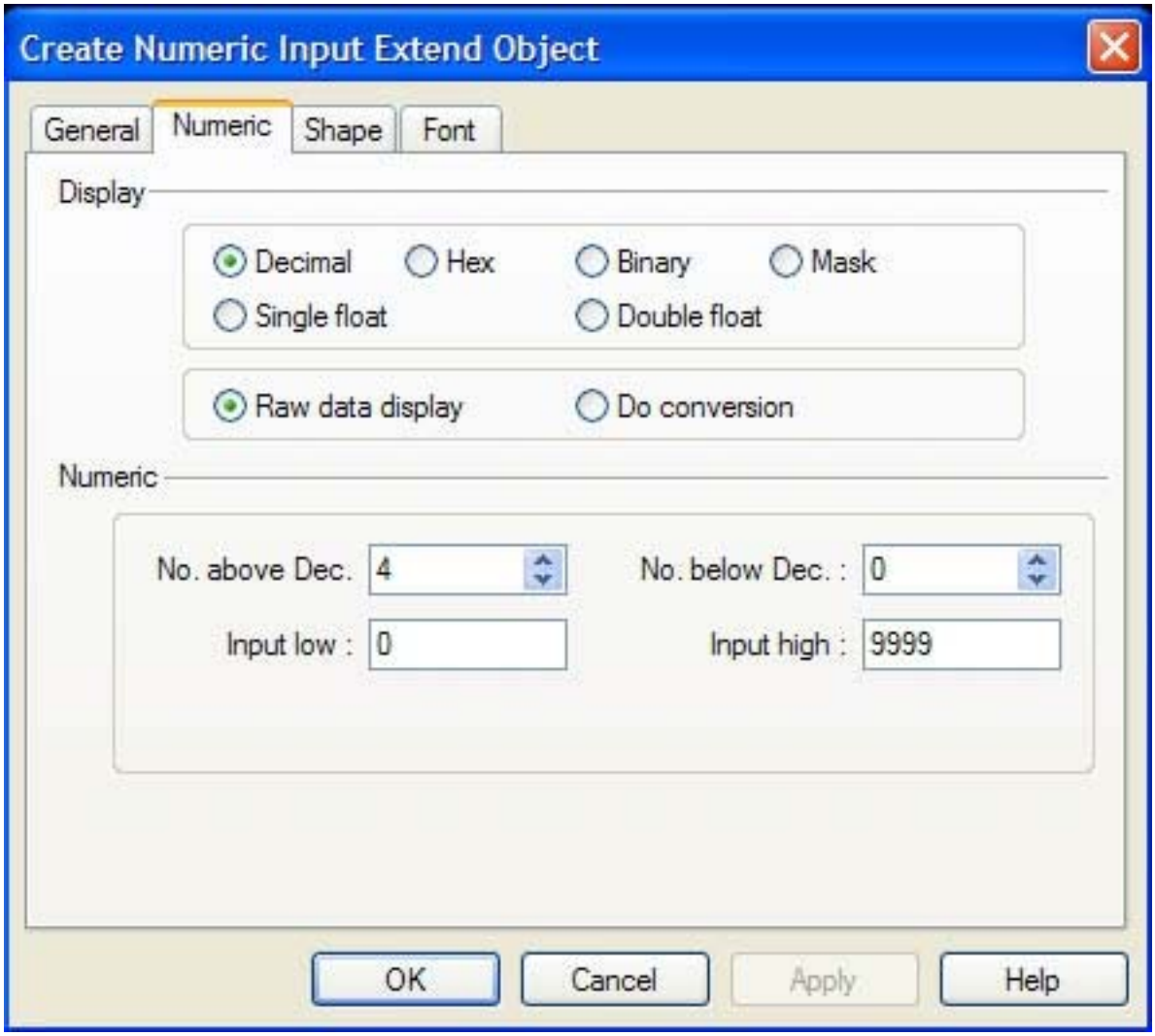
To create a Numeric Input Extend Object 

1. From the **Parts** menu, click **Numeric Input Extend**. Or click the **Numeric Input Extend** icon in the Part1 toolbar. The Create Numeric Input Extend Object dialog box appears.



2. Use the **Description:** box to enter a title for the Numeric Input part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC register or HMI internal memory address that the part continuously reads and writes to. You can select **BIN** (binary) or **BCD** format. Select **No. of words:** that the part should read, (1= 16 bits, 2= 32 bits, 4= 64 bits).
4. In the **Trigger address** frame, select the PLC coil or HMI internal memory address that is used to trigger write access to the register. If the coil is set, then write access is enabled. If the coil is clear, then write access is disabled. Note: If you want the register to always have write access, use one of the 'always on' internal coils of the HMI, (LB9000-9009).

5. Click the **Numeric** tab to display the Numeric form.



6. In the **Display** attribute box, select the format type you wish to use. The options are:

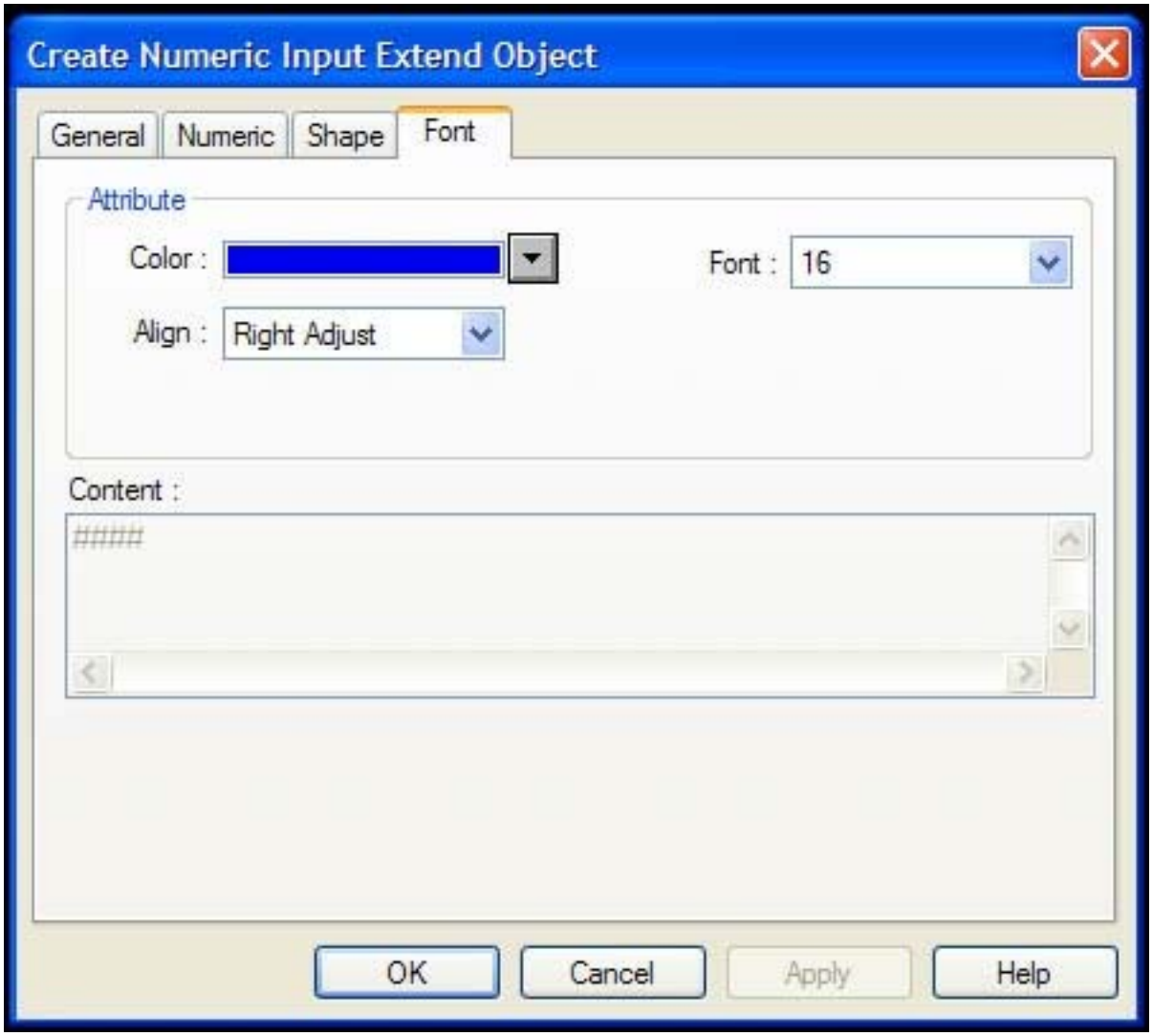
Format	Range	No. of Words	Description
Decimal	-32768 to +32767	1	signed 16 bit format
	-999999999 to +2147483647	2	signed 32 bit format
Hex	0000 to FFFF	1	hexadecimal 16 bit format
	00000000 to FFFFFFFF	2	hexadecimal 32 bit format
Binary	00… to 11… (16 bits)	1	binary 16 bit format
	00… to 11… (32 bits)	2	binary 32 bit format
Mask	*****	1,2	masks the value with asterisks
Single Float	-999999999.9999999999 to 9999999999.9999999999	2	IEEE 754 format; single precision
Double Float	-999999999.9999999999 to 9999999999.9999999999	4	IEEE 754format; double precision

7. If **Decimal** format is selected, then you have the option of selecting **Raw data display** or **Do conversion**. Use **Raw data display** when you wish to display the true numeric value of the register. Select **Do conversion** when you wish to scale the value.
8 In the **Display** attribute box, select the format type you wish to use. The options are:

Format	Options	Range	Description
Decimal (with Raw data display)	No. above Dec.:	1-10	specifies number of digits to display (for example, 2 digits displays -9 to 99); any number that cannot be displayed with the specified number of digits is represented with asterisks.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.
	Input low:	-2147483648 to +2147483647	This is the lowest data value allowed when the HMI operator enters a number.
	Input high:	-2147483648 to +2147483647	This is the highest data value allowed when the HMI operator enters a number.
Decimal (with Do conversion)	No. above Dec.:	1-10	specifies number of digits to the left of the decimal point.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.
	Input low:	-2147483648 to +2147483647	This is the lowest raw data value that is in the PLC data register
	Input high:	-2147483648 to +2147483647	This is the highest raw data value that is in the PLC data register
	Engineering low:	-2147483648 to +2147483647	This is the lowest data value that you would like displayed on the HMI and the minimum value the HMI operator is allowed to enter.
	Engineering high:	-2147483648 to +2147483647	This is the highest data value that you would like displayed on the HMI and the maximum value the HMI operator is allowed to enter.
Hex	No. above Dec.:	Not Apply (NA)	4 digits are used for 16 bit; 8 digits for 32 bit.
	Input low:	0 to $+2147483647_d$	This is a *decimal* number which the HMI converts to hexadecimal for establishing the lowest value that the HMI operator is allowed to enter. For example, if you want the minimum hex value to be 100n, then you would enter value 256d.
	Input high:	0 to $+2147483647_d$	This is a *decimal* number which the HMI converts to hexadecimal for establishing the highest value that the HMI operator is allowed to enter. For example, if you want the maximum value to be 1000h, then you would enter the value 4096d.
Binary	No. above Dec.:	NA	16 digits are used for 16 bit; 32 digits for 32 bit.
	Input low:	0 to $+2147483647_d$	This is a decimal number which the HMI converts to binary for establishing the lowest value that the HMI operator is allowed to enter. For example, if you want the minimum binary value to be 1001b, then you would enter value 9d.
	Input high:	0 to $+2147483647_d$	This is a decimal number which the HMI converts to binary for establishing the highest value that the HMI operator is allowed to enter. For example, if you want the maximum binary value to be 10100101b, then you would enter value 165d
Mask	No. above Dec.:	1-10	The number of asterisks to display. When HMI operator is entering a new number, only asterisks are displayed as the number is entered. Decimal format is used when Mask is selected.
	Input low:	-2147483648 to +2147483647	This is the lowest data value allowed when the HMI operator enters a number.
	Input high:	-2147483648 to +2147483647	This is the highest data value allowed when the HMI operator enters a number.

Single Float	No. above Dec.:	1-10	specifies number of digits to the left of the decimal point.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.
	Input low:	-2147483648 to +2147483647	This is the lowest data value allowed when the HMI operator enters a number.
	Input high:	-2147483648 to +2147483647	This is the highest data value allowed when the HMI operator enters a number.
Double Float	No. above Dec.:	1-10	specifies number of digits to the left of the decimal point.
	No. below Dec.:	0-10	specifies number of digits to the right of the decimal point.
	Input low:	-2147483648 to +2147483647	This is the lowest data value allowed when the HMI operator enters a number.
	Input high:	-2147483648 to +2147483647	This is the highest data value allowed when the HMI operator enters a number.

9. Click the **Shape** tab to configure a shape or bitmap that is displayed behind the number.
10. Click the **Font** tab to display the Font form.



11. Enter the color that you want for the digits displayed in the **Color:** box.
12. Enter how you want the digits to be positioned in the **Align:** box. The choices are:

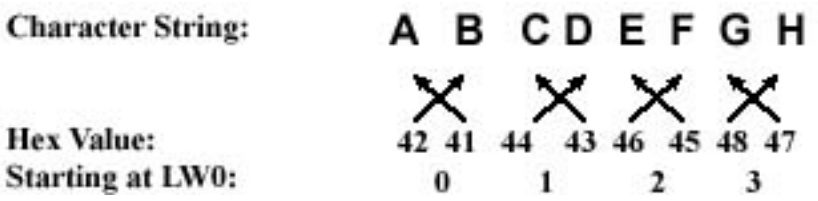
- **Right Adjust** - aligns the number to the right margin of the part.
- **Left Adjust** - aligns the number to the left margin of the part.
- **Leading Zero** - inserts leading zeroes

13. Enter the size of the digits in the **Font:** box.
14. Click **OK**. The Numeric Input Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
15. Once the part is placed onto the window, you can modify the attributes by double-clicking on the part to display the Numeric Input Object's Attribute dialog box. You may also resize the object by single clicking on the object and pulling the border of the object to the appropriate size. The object's associated text may be repositioned by clicking on the object, and then by dragging the text to it's new position.

The ASCII Data Object

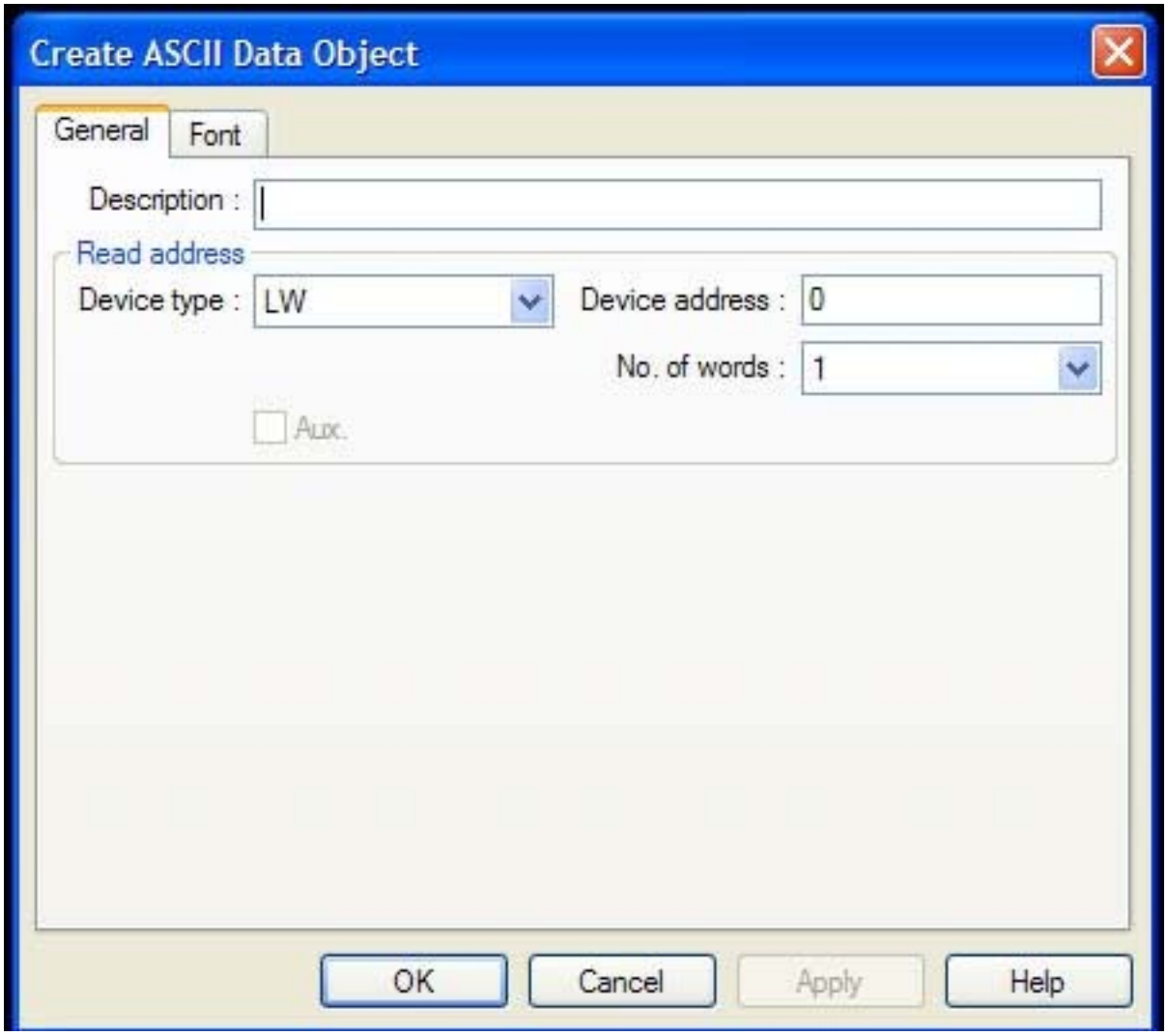
The ASCII Data Object is used to display the character values of a series of PLC registers. The object continuously reads the PLC registers and displays the corresponding ASCII Data in the format specified. The ASCII Data Object can read up to 16 PLC registers to display a maximum of 32 characters.

Each 16 bit register stores two ASCII characters. The first ASCII character is stored in the eight least significant bits of the register. The 16-bit register that you select is the starting point for the ASCII character string. The example below illustrates how an eight character string is arranged:



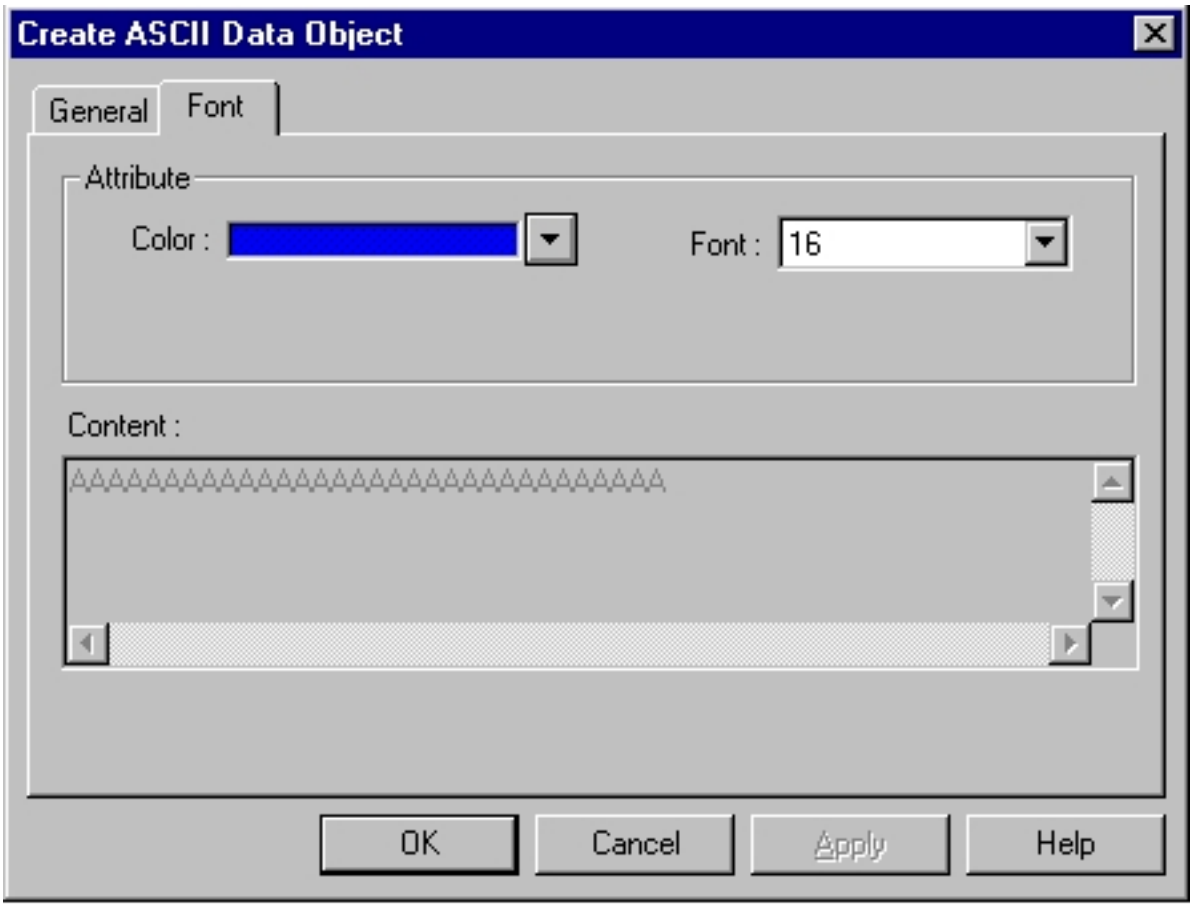
To create a ASCII Data Object 

1. From the **Parts** menu, click **ASCII Data**. Or click the **ASCII Data** icon in the Part1 toolbar. The Create ASCII Data Object dialog box appears.



2. Use the **Description:** box to enter a title for the ASCII Data part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC register or HMI internal memory address that the part continuously reads. Select **No. of words:** that the part should read, (1= 2 characters, Range: 1 to 16).

4. Click the **Font** tab to display the Font form.



5. Enter the color that you want for the characters displayed in the **Color:** box.
6. Enter the size of the digits in the **Font:** box.
7. Click **OK**. The ASCII Data Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
8. Once the part is placed onto the window, you can modify the attributes by double-clicking on the part to display the ASCII Data Object's Attribute dialog box.

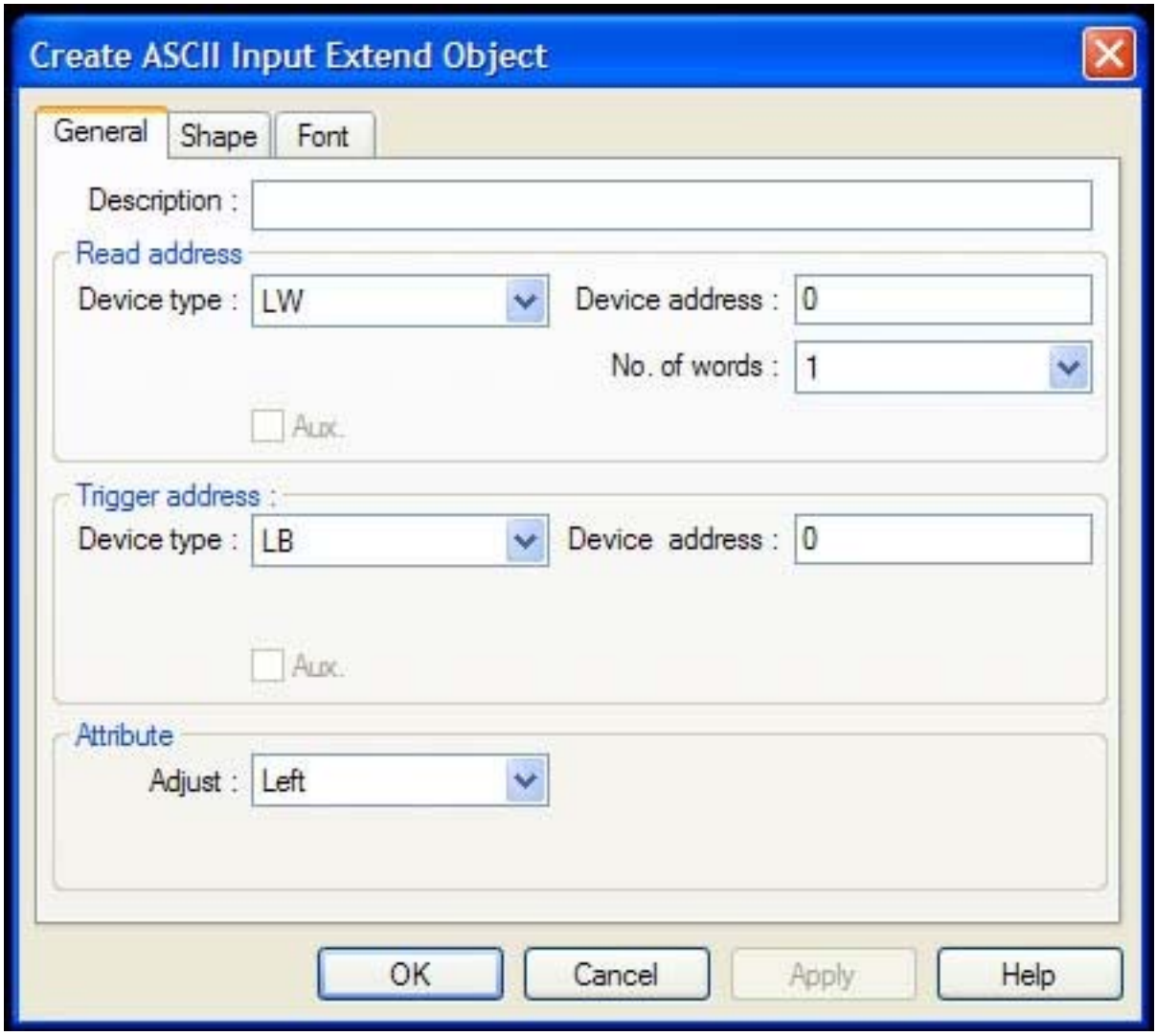
The ASCII Input Extend Object

The ASCII Input Extend Object is used to enter characters into a series of PLC registers. The object continuously reads the PLC registers and displays the corresponding ASCII characters in the format specified. The ASCII Input Extend Object can write a maximum of 32 characters to 16 contiguous PLC registers.

See the ASCII Data section for information on how the ASCII characters are actually stored in the PLC registers.

▶ **To create a ASCII Input Extend Object**

1. From the **Parts** menu, click **ASCII Input Extend**. Or click the **ASCII Input Extend** icon in the Part1 toolbar. The Create ASCII Input Extend Object dialog box appears.

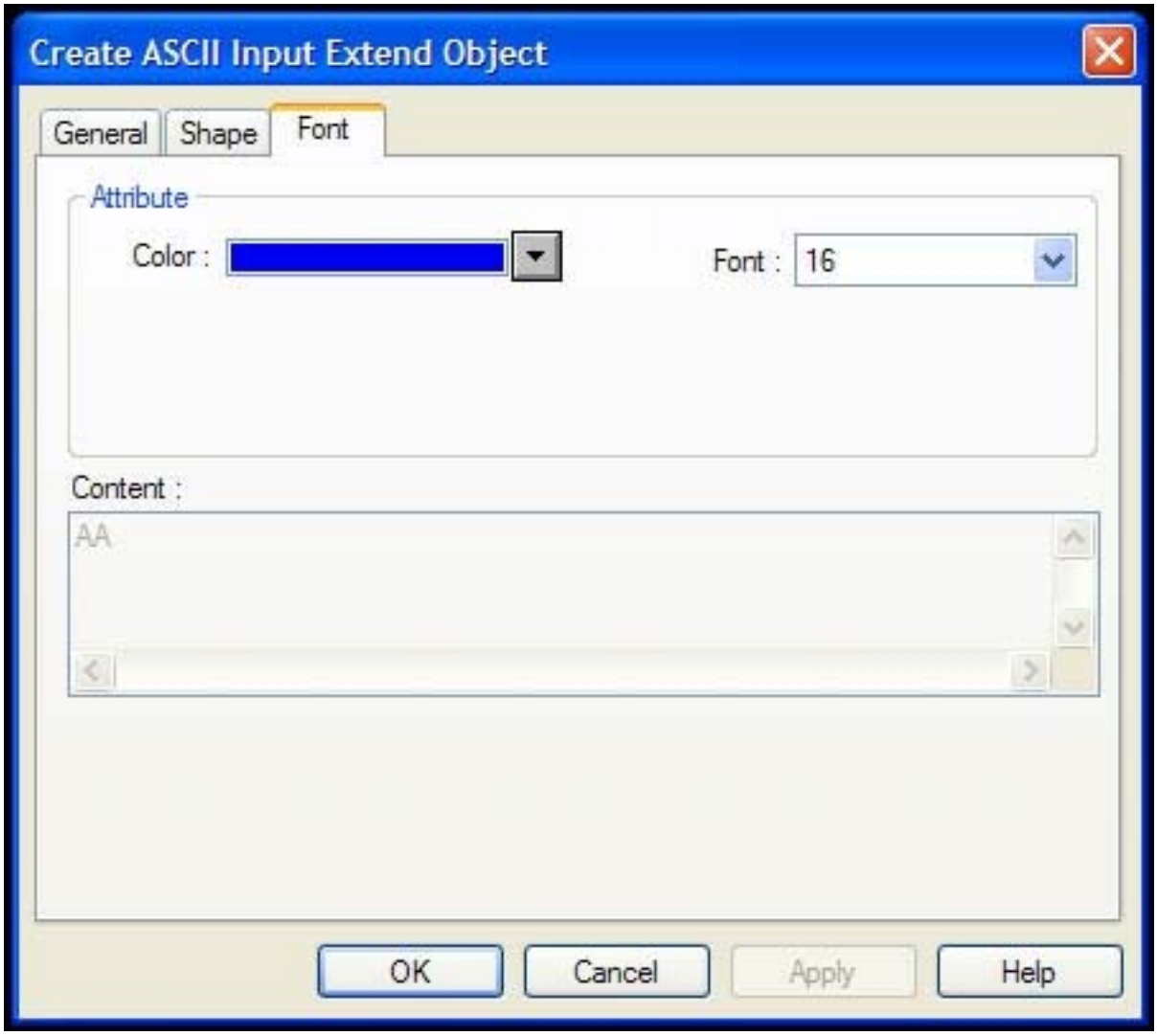


2. Use the **Description:** box to enter a title for the ASCII Input part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC register or HMI internal memory address that the part continuously reads. Select **No. of words:** that the part should read, (ex.1= 2 characters, Range: 1 to 16).
4. In the **Trigger address** frame, select the PLC coil or HMI internal memory address that is used to trigger write access to the register. If the coil is set, then write access is enabled. If the coil is clear, then write access is disabled. Note: If you want the register to always have write access, use one of the 'always on' internal coils of the HMI, (LB9000-9009).

5. Enter how you want the digits to be positioned in the **Adjust:** box. The choices are:

- Left - the HMI enters the characters into the PLC registers so that when the HMI reads the contents, the characters are left aligned.
- **Right** - the HMI enters the characters into the PLC registers so that when the HMI reads the contents, the characters are right aligned.

6. Click the **Shape** tab to configure a shape or bitmap that is displayed behind the number.
7. Click the **Font** tab to display the Font form.



8. Enter the color that you want for the characters displayed in the **Color:** box.
9. Enter the size of the digits in the **Font:** box.
10. Click **OK**. The ASCII Input Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
11. Once the part is placed onto the window, you can modify the attributes by double-clicking on the part to display the ASCII Input Extend Object's Attribute dialog box. You may also resize the object by single clicking on the object and pulling the border of the object to the appropriate size. The object's associated text may be repositioned by clicking on the object, and then by dragging the text to it's new position.

The Moving Shape Object

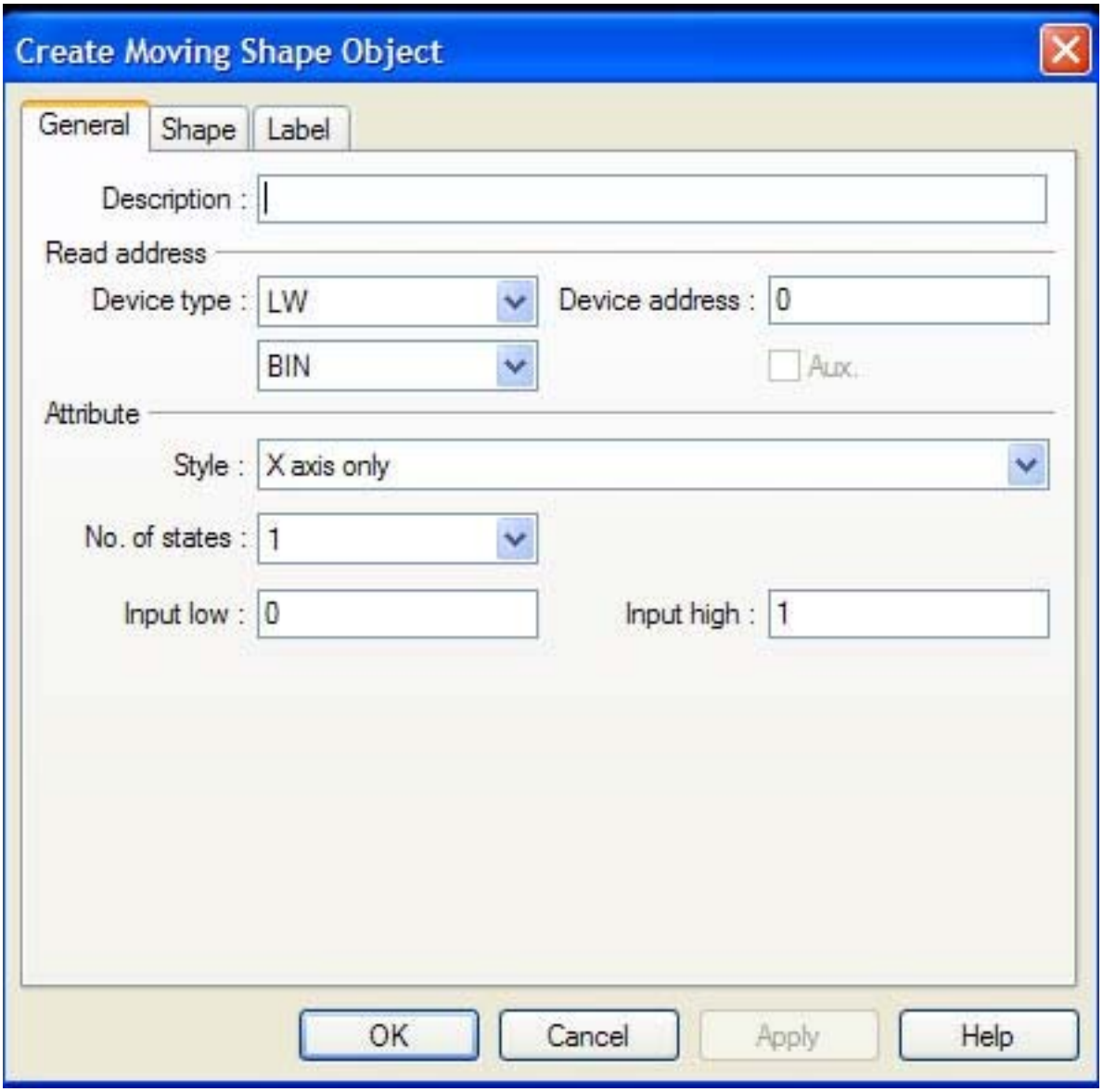
The Moving Shape Object performs three functions depending upon the values in three consecutive PLC registers:

- Similar to the Word Lamp Object, the Moving Shape object continuously reads a PLC register and displays the corresponding shape or bitmap that is tagged to the number of states defined.
- the Moving Shape object continuously reads a second PLC register to determine where along an x-axis to display the shape or bitmap.
- the Moving Shape object continuously reads a third PLC register to determine where along a y-axis to display the shape or bitmap.

This feature allows you to create a shape or bitmap on a screen that can change the way it looks as well as where it is located on the HMI screen depending upon the control of the PLC. This can be used to simulate a control process such as some products moving on a conveyor system to different parts of the plant.

To create a Moving Shape Object 

1. From the **Parts** menu, click **Moving Shape**. Or click the **Moving Shape** icon in the Part2 toolbar. The Create Moving Shape Object dialog box appears.



2. Use the **Description:** box to enter a title for the Moving Shape part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the starting PLC coil or HMI internal memory address that the part reads from. You can select **BIN** (binary) or **BCD** format. The number of registers read depends upon the Style used.

4. In the **Attribute** frame, select how you want the Moving Shape object to operate:

Style	Option	Description
X axis only		Reads two 16-bit registers. Register #1 determines the state of the shape displayed. Register #2 determines the position of the shape along the X axis. The HMI places the shape relative to the starting location. For example, if the shape is located on the window at X=50, and Register #2 has a value of 10, the shape is placed at X=60. If Register #2 has a value of -10, the shape is placed at X=40.
	No. of states:	Range = 1 to 32
	Input low:	Not used
	Input high:	Not used
Y axis only		Reads two 16-bit registers. Register #1 determines the state of the shape displayed. Register #2 determines the position of the shape along the Y axis. The HMI places the shape relative to the starting location. For example, if the shape is located on the window at Y=50, and Register #2 has a value of 10, the shape is placed at Y=60. If Register #2 has a value of -10, the shape is placed at Y=40.
	No. of states:	Range = 1 to 32
	Input low:	Not used
	Input high:	Not used
X & Y axis		Reads three 16-bit registers. Register #1 determines the state of the shape displayed. Register #2 determines the position of the shape along the X axis. Register #3 determines the position of the shape along the Y axis. The HMI places the shape relative to the starting location. For example, if the shape is located on the window at X=50, Y=10, and Register #2 has a value of 10 and Register #3 has a value of 5, the shape is placed at X=60, Y=15. If Register #2= -10 and Register #3= -5, the shape is placed at X=40, Y=5.
	No. of states:	Range = 1 to 32
	Input low:	Not used
	Input high:	Not used
X axis w/scaling		Reads two 16-bit registers. Register #1 determines the state of the shape displayed. Register #2 determines the position of the shape along the X axis. The HMI places the shape relative to the starting location using scaling to calculate position. For example, if Input low=0, Input high=10, Scaling low=0, and Scaling high=100, then when the value in the PLC register increases by 1, the shape moves *right* along the X axis by 10 pixels.
	No. of states:	Range = 1 to 32
	Input low:	This is the lowest raw data value that is in the PLC data register; Range = -32767 to +32766
	Input high:	This is the highest raw data value that is in the PLC data register; Range = -32766 to +32767
	Scaling low:	This is the low scaled value.
	Scaling high:	This is the high scaled value.
Y axis w/scaling		Reads two 16-bit registers. Register #1 determines the state of the shape displayed. Register #2 determines the position of the shape along the Y axis. The HMI places the shape relative to the starting location using scaling to calculate position. For example, if Input low=0, Input high=10, Scaling low=0, and Scaling high=100, then when the value in the PLC register increases by 1, the shape moves *down* along the Y axis by 10 pixels.
	No. of states:	Range = 1 to 32
	Input low:	This is the lowest raw data value that is in the PLC data register; Range = -32767 to +32766
	Input high:	This is the highest raw data value that is in the PLC data register; Range = -32766 to +32767
	Scaling low:	This is the low scaled value.
	Scaling high:	This is the high scaled value.

X axis w/reverse scaling		Reads two 16-bit registers. Register #1 determines the state of the shape displayed. Register #2 determines the position of the shape along the X axis. The HMI places the shape relative to the starting location using scaling to calculate position. For example, if Input low=0, Input high=10, Scaling low=0, and Scaling high=100, then when the value in the PLC register increases by 1, the shape moves *left* along the X axis by 10 pixels.
	No. of states:	Range = 1 to 32
	Input low:	This is the lowest raw data value that is in the PLC data register; Range = -32767 to +32766
	Input high:	This is the highest raw data value that is in the PLC data register; Range = -32766 to +32767
	Scaling low:	This is the low scaled value.
	Scaling high:	This is the high scaled value.
Y axis w/reverse scaling		Reads two 16-bit registers. Register #1 determines the state of the shape displayed. Register #2 determines the position of the shape along the Y axis. The HMI places the shape relative to the starting location using scaling to calculate position. For example, if Input low=0, Input high=10, Scaling low=0, and Scaling high=100, then when the value in the PLC register increases by 1, the shape moves *up* along the Y axis by 10 pixels.
	No. of states:	Range = 1 to 32
	Input low:	This is the lowest raw data value that is in the PLC data register; Range = -32767 to +32766
	Input high:	This is the highest raw data value that is in the PLC data register; Range = -32766 to +32767
	Scaling low:	This is the low scaled value.
	Scaling high:	This is the high scaled value.

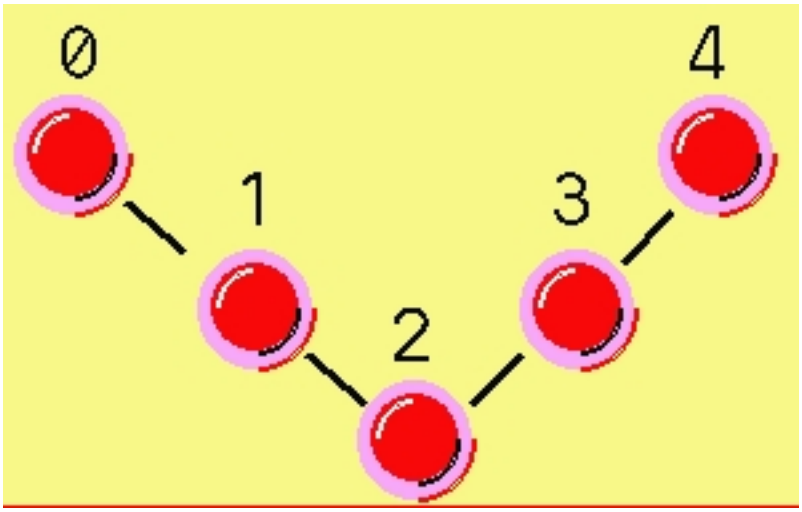
5. Refer to the Bit Lamp Object section for information on using the **Shape** tab.
6. Refer to the Bit Lamp Object section for information on using the **Label** tab.
7. Click **OK**. The Create Moving Shape Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
8. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Click on the label again. Now only the label is highlighted, allowing you to move it without moving the part.
   Note: if you double-click (click twice rapidly) then you will not highlight the label but rather

The Animation Object

The Animation Object is very similar to the Moving Shape Object. It also allows you to create a shape or bitmap on a screen that can change the way it looks as well as where it is located on the HMI screen depending upon the control of the PLC. However, the Animation Object moves along a predefined path that is determined when you create the part. Because of this, the Animation Object requires only two registers for changing state and X/Y movement:

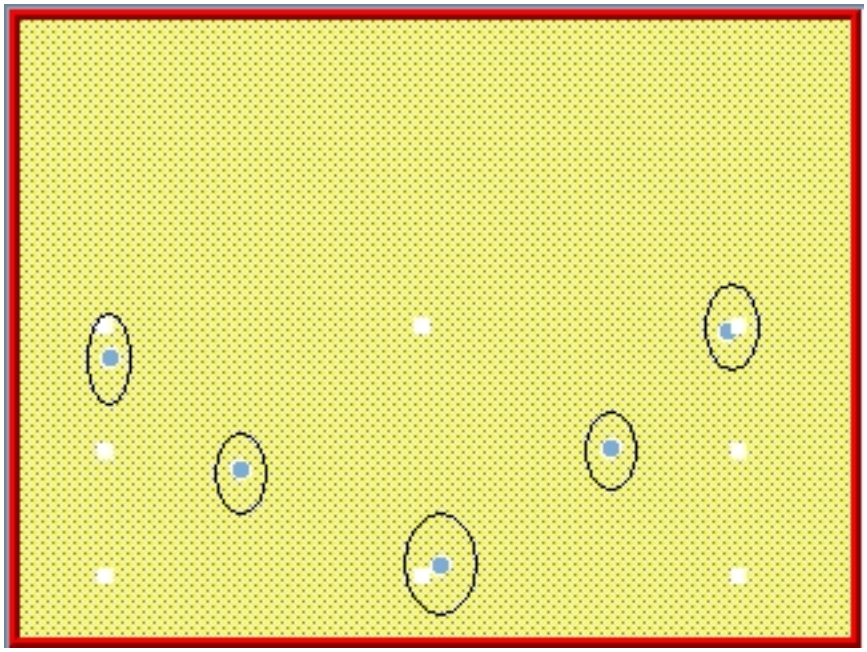
- the Animation object continuously reads a PLC register and displays the corresponding shape or bitmap that is tagged to the number of states defined.
- the Animation object continuously reads a second PLC register to determine which position along the predefined path it should be.

As shown in the following example, the part moves along a predefined track when the value in the second PLC register is incremented:

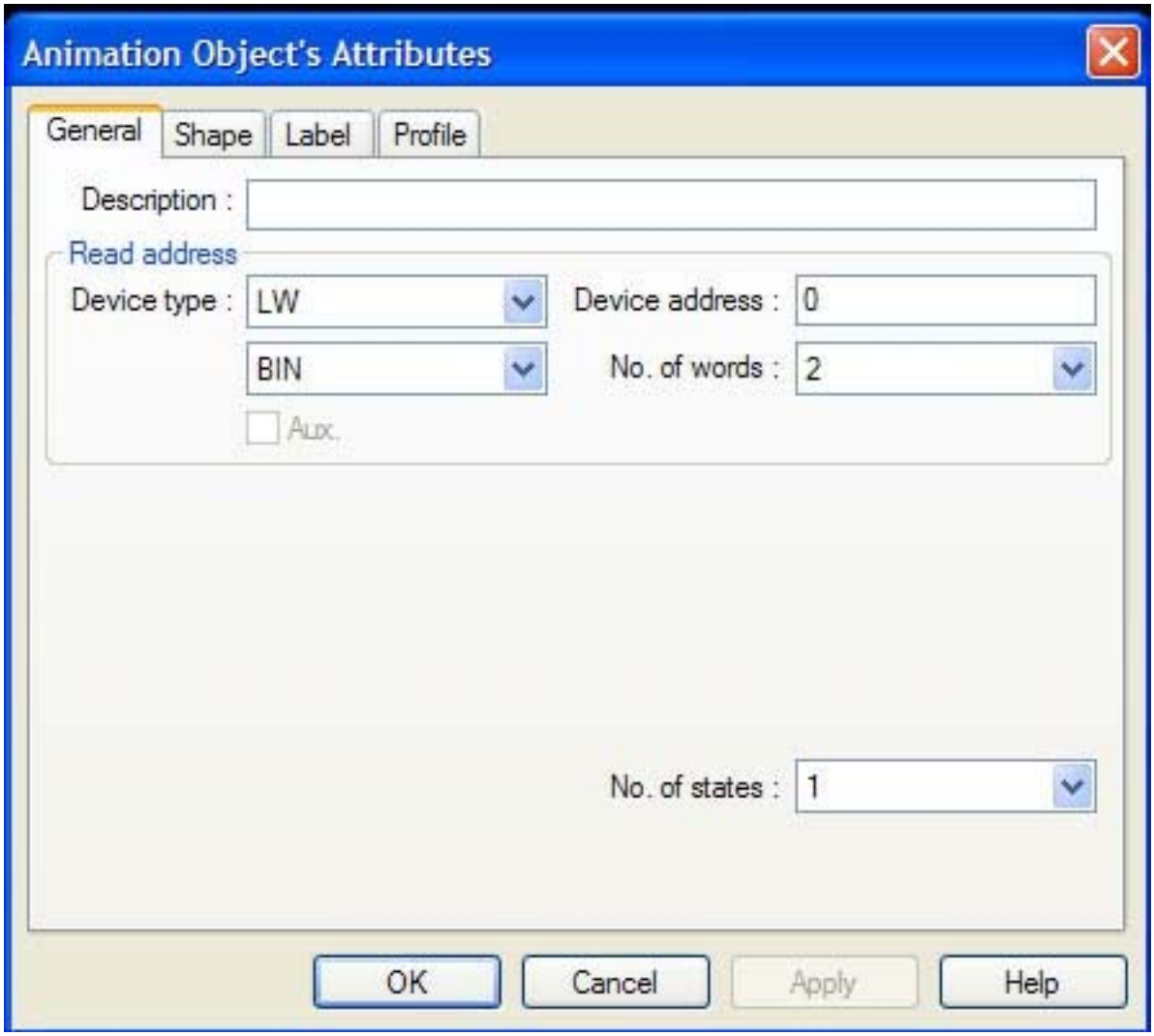


To create an Animation Object 

1. From the **Parts** menu, click **Animation**. Or click the **Animation** icon in the Part2 toolbar. The mouse cursor changes from a pointer to a crosshair when placed over the work area of EasyBuilder. Use the cursor to set the path that the Animation object is to take. At each location along the path, click the mouse to place another position marker. A maximum of 255 position markers may be placed onto the window screen.
2. When you have finished placing the position markers, right click the mouse to display the part at the first position.

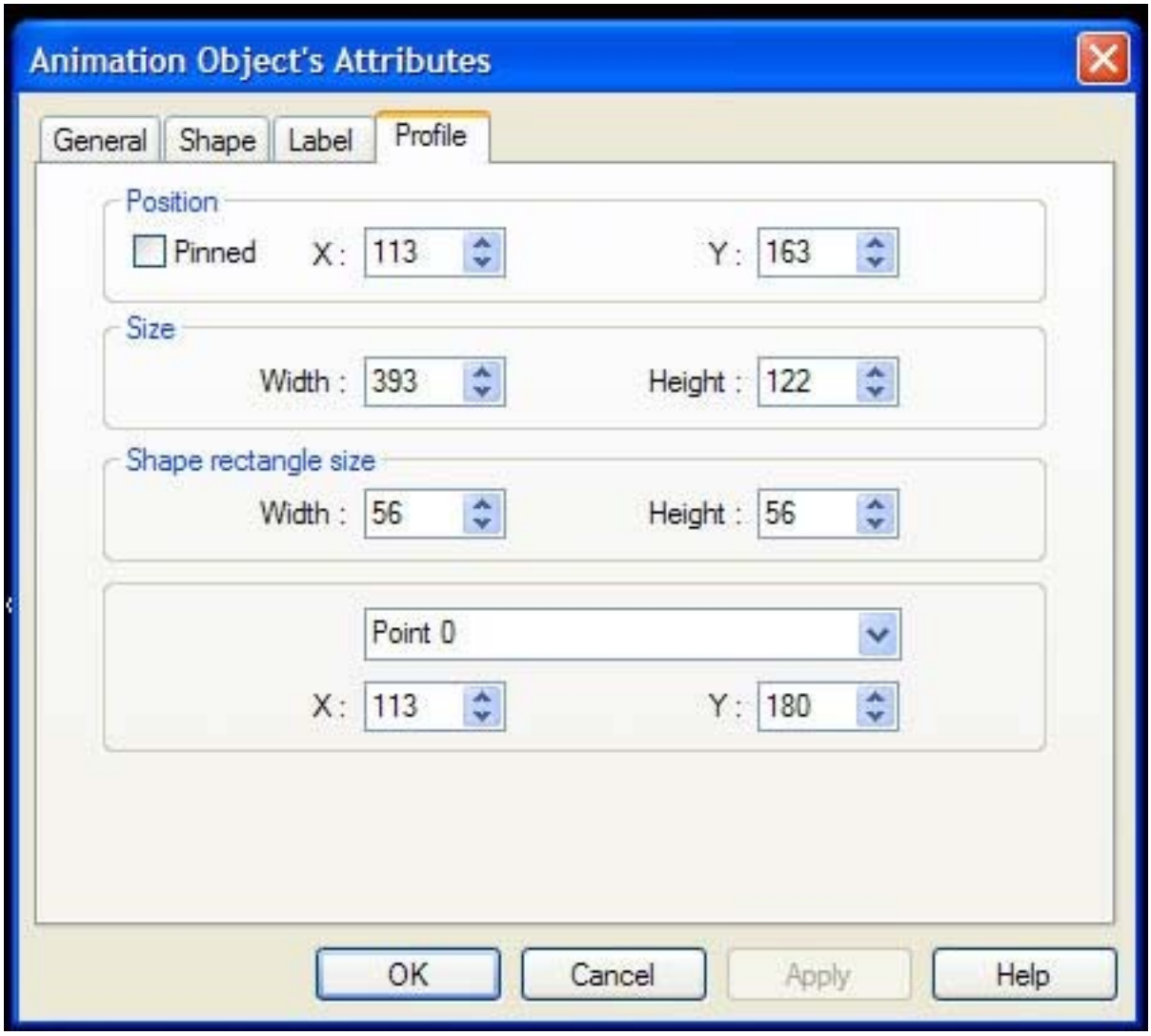


3. Double click the part. The Animation Object's Attribute dialog box appears.



4. Use the **Description:** box to enter a title for the Animation part. A description is not necessary but does help you identify the purpose of the part.
5. the **Read address** frame, select the starting PLC coil or HMI internal memory address that the part reads from. You can select **BIN** (binary) or **BCD** format. The **No. of words:** is set to 2 and cannot be changed.
6. In the **No. of states:** box, select the total number of states to be displayed.
7. Refer to the Bit Lamp Object section for information on using the **Shape** tab.
8. Refer to the Bit Lamp Object section for information on using the **Label** tab.

9. Click on the **Profile** tab. The Profile dialog box appears.



10. The **Position** settings refer to the overall location of the Animation Object on the window screen.
11. The **Size** settings refer to the overall size of the Animation Object.
12. The **Shape rectangle size** settings refer to the size of the shape or bitmap that you are using.
13. You can also manipulate the position of each point along the path of movement. Click on the points pull-down box to select the point you wish to change, then edit the X and Y positions.
14. Click **OK**. The Animation Object's Attribute form closes and the main screen of EasyBuilder appears with the Animation part you just created on the screen. The shape you selected is located at the first position on the path.
15. Once the part is placed onto the window, you can adjust the attributes by double-clicking on the part to enter the Animation Object's Attribute dialog box.

The next chapter shows how to construct a keypad for editing PLC data registers.

Chapter 9 - Using and Creating Keypads

The HMI operator must have a keypad available to enter new data when using the Numeric Input Object or the ASCII Input Object. EasyBuilder includes three group libraries (keypad.glb, user-library-keypad.glb and user-library-keypad1.glb), which contain sample keypads that can be placed onto a window screen. You can also create your own custom keypads. This chapter focuses on how to create keypads, display them on the window screen, and use them for entering data into PLC registers.

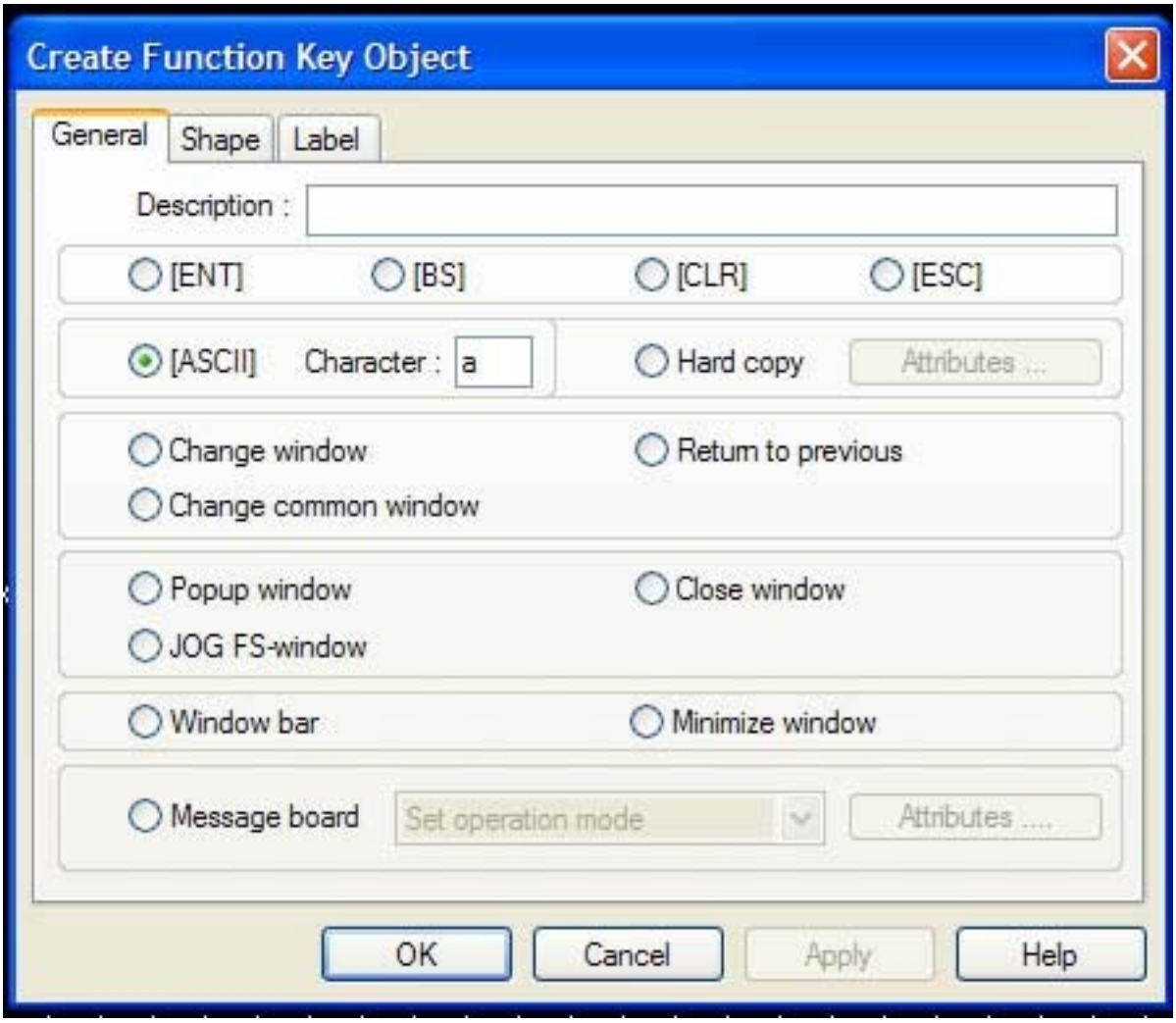
How to Create a Keypad

Any keypad that you create is actually composed of keys using the Function Key Object. The Function Key Object can create alphanumeric keys as well as control keys, (i.e. Delete, Enter, Clear, etc.).

To create a Function Key Object 

1. From the **Parts** menu, click **Function Key**. Or click the **Function Key** icon in the Part1 toolbar. The Create Function Key Object dialog box appears.
2. Use the **Description:** box to enter a title for the Function Key part. A description is not necessary but does help you identify the purpose of the part.

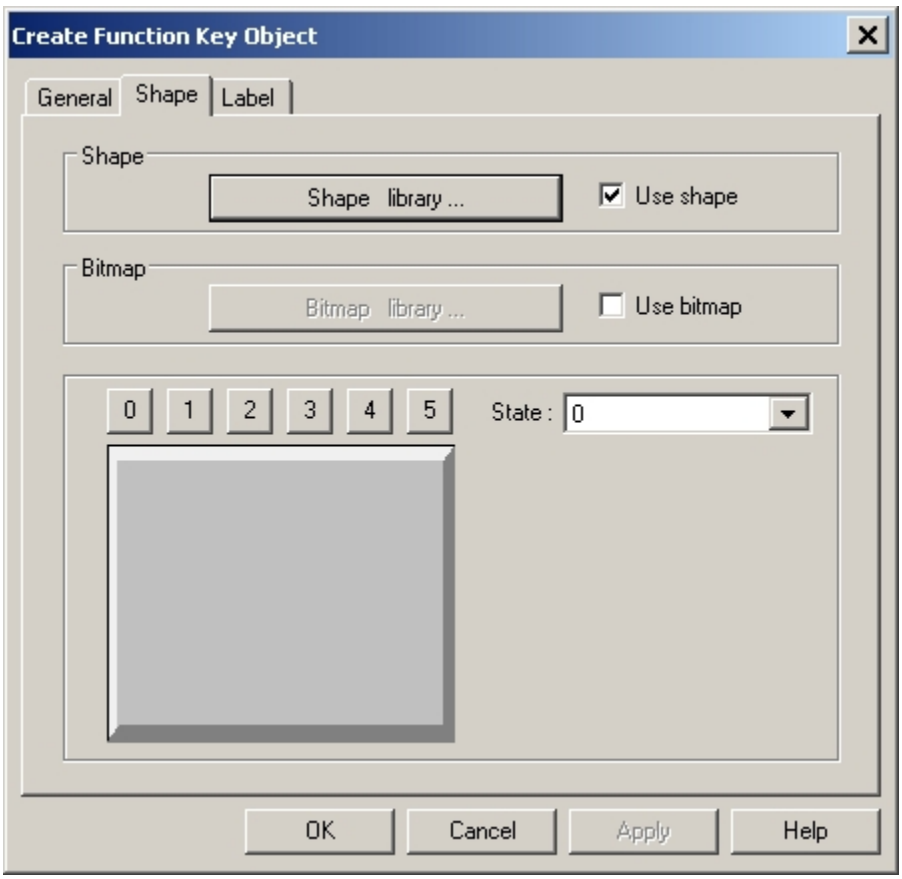
3. Just below the **Description:** box, are four control key options. Control keys are used during data entry to perform a specific function.



Function	Description
[ENT]	Configures the Function Key as an Enter key. When pressed, it will write the alphanumeric characters entered into a Numeric Input Object or ASCII Input Object to the target PLC register.
[BS]	Configures the Function Key as a Delete key. When pressed, it will delete the last alphanumeric character entered.
[CLR]	Configures the Function Key as a Clear key. When pressed, it will clear the data displayed in the Numeric Input Object or ASCII Input Object. Note: this key does not clear the actual target PLC register until the Enter key is pressed.
[ESC]	Configures the Function Key as an Escape key. When pressed, it will exit the editing mode.

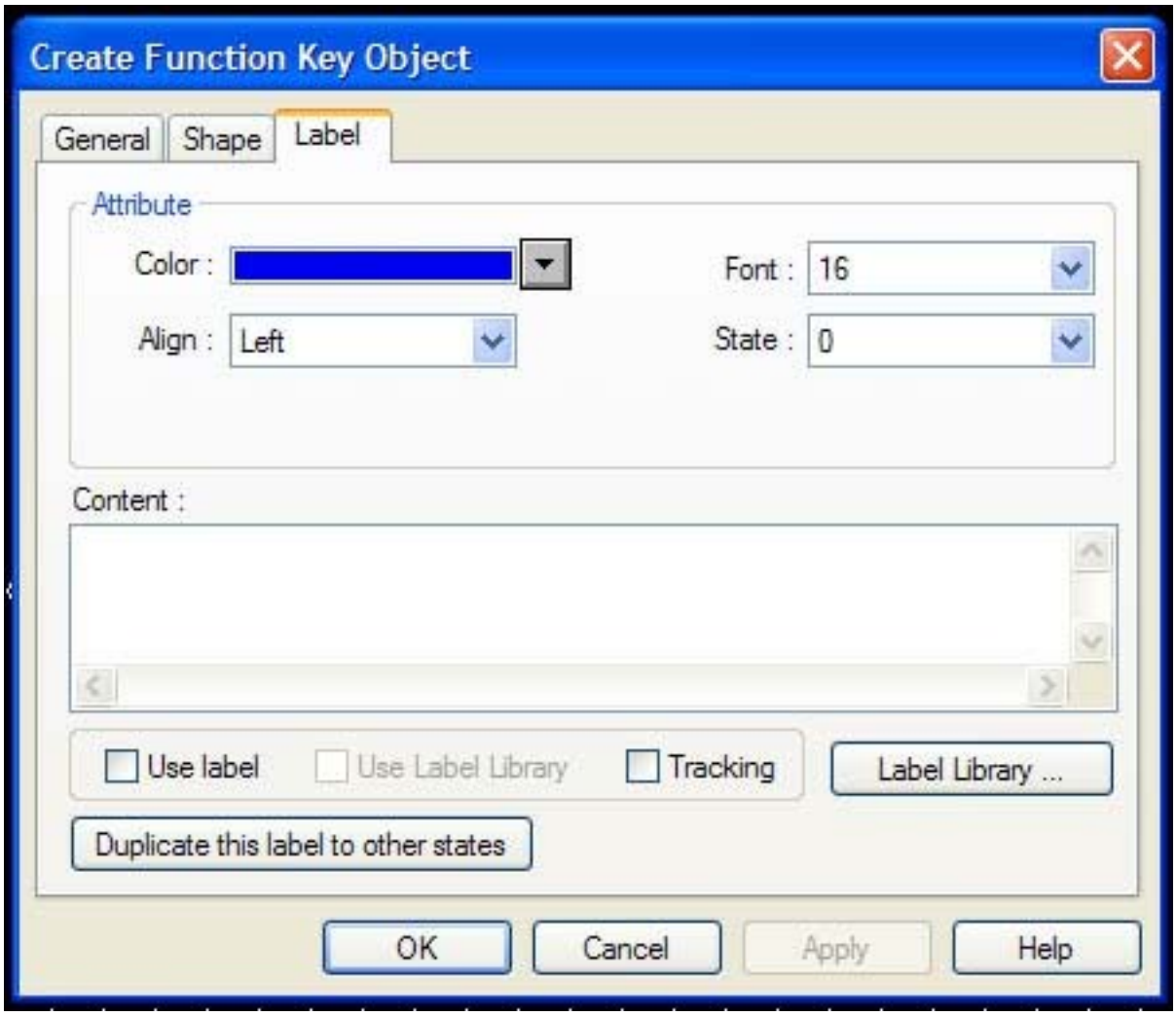
4. Just below the four control key options, is the **ASCII** checkbox. Select this when you want to configure the Function Key to enter an alphanumeric character. When this checkbox is selected, the **Character:** box is activated. Use this box to enter the ASCII character you want to use.
5. The **Hard Copy** checkbox is for a printer option, which is not currently supported.

6. Click the **Shape** tab to display the Shape form.



7. Select either a shape or bitmap to represent the Function Key object. If you need more information on how to do this, consult Chapter 6 “Creating Graphics Objects”.

8. Click the **Label** tab to display the Label form.



9. Click the **Use Label** checkbox to use a label.
10. You can have a different label for each state of the Bit Lamp. In the **State:** box, enter the state that you wish to edit, then enter the label for that state in the **Content:** box. Do the same for the other state.
11. Enter the color that you want for the label in the **Color:** box. You can select a different color for each label state.
12. Enter how you want the label to be positioned in the **Align:** box. You can select a different position for each label state.
13. Finally, enter the size of the label in the **Font:** box. You can select a different font for each label state.
14. Click the **Tracking** checkbox if you want the labels for all states to follow or 'track' with the movement of one label when it is moved after the Function Key object is placed onto the window. Click **Duplicate this label to other states** if you want to use the same text on all states.
15. Click **OK**. The Create Function Key Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
16. Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object. Now click on the label again. Now only the label is highlighted, allowing you to move it without moving the

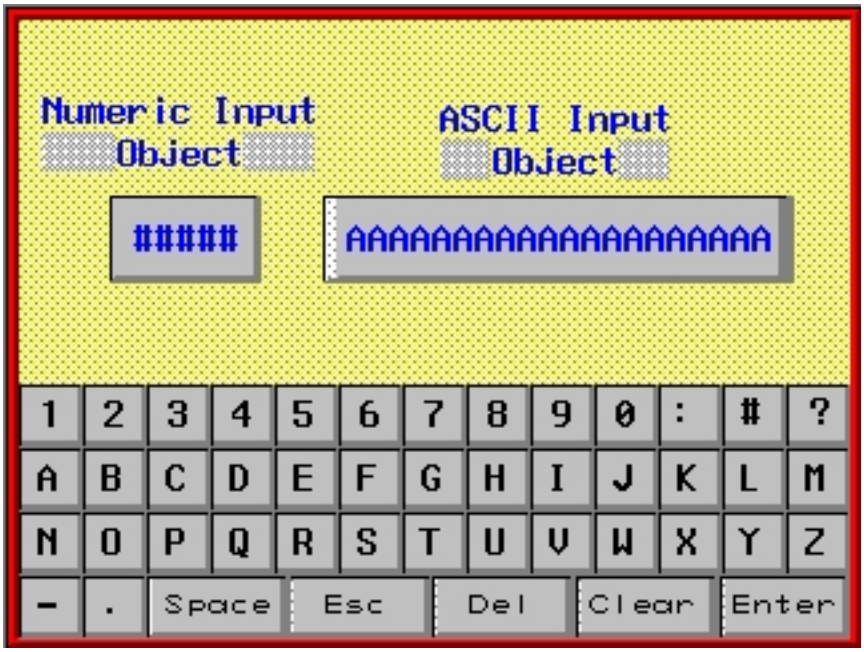
part. Note: if you double-click (click twice rapidly) then you will not highlight the label but rather enter the Function Key Object's Attribute dialog box.

To create a custom keypad, you create several of these Function Key Objects and group them together on a window to form the keypad. Following the procedure outlined in Chapter 6, you can then save the keypad to a group library for later use in other projects. Several predefined keypads are included with EasyBuilder. To use these keypads, use the **Call Group Library** function and access the **keypad** library.

Displaying and Using a Keypad

Let's go through a sample project that uses one of the predefined keypads included in EasyBuilder to edit a Numeric Input Object and an ASCII Input Object. This example will show you how to place these objects on a window screen, and then use the keypad to edit the data registers.

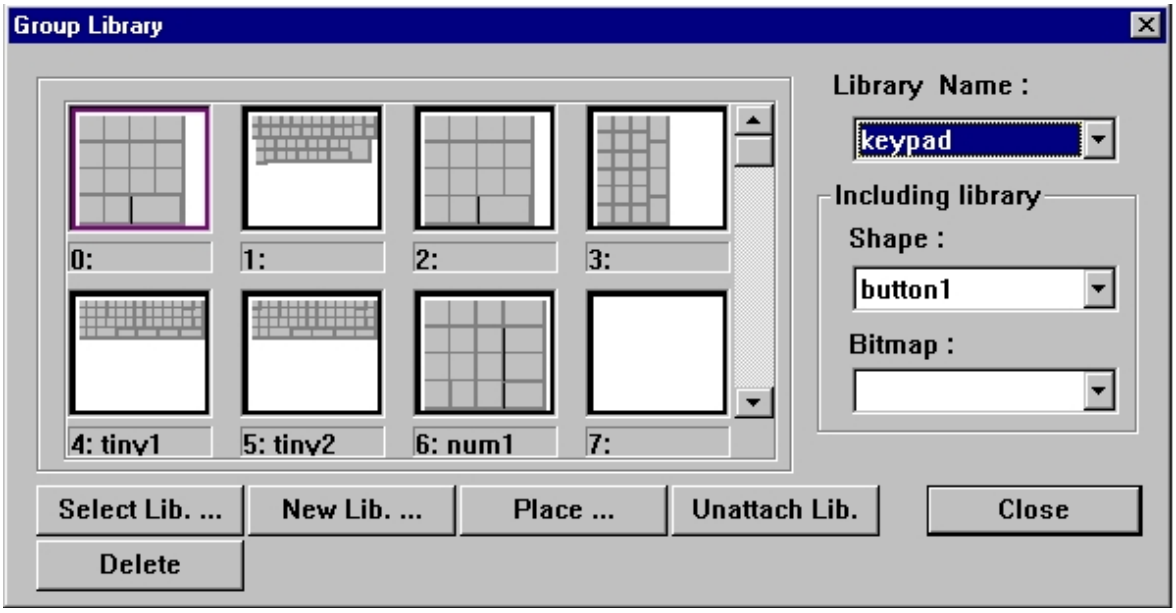
The screen we create should look like the following when finished:



To create a sample keypad entry screen

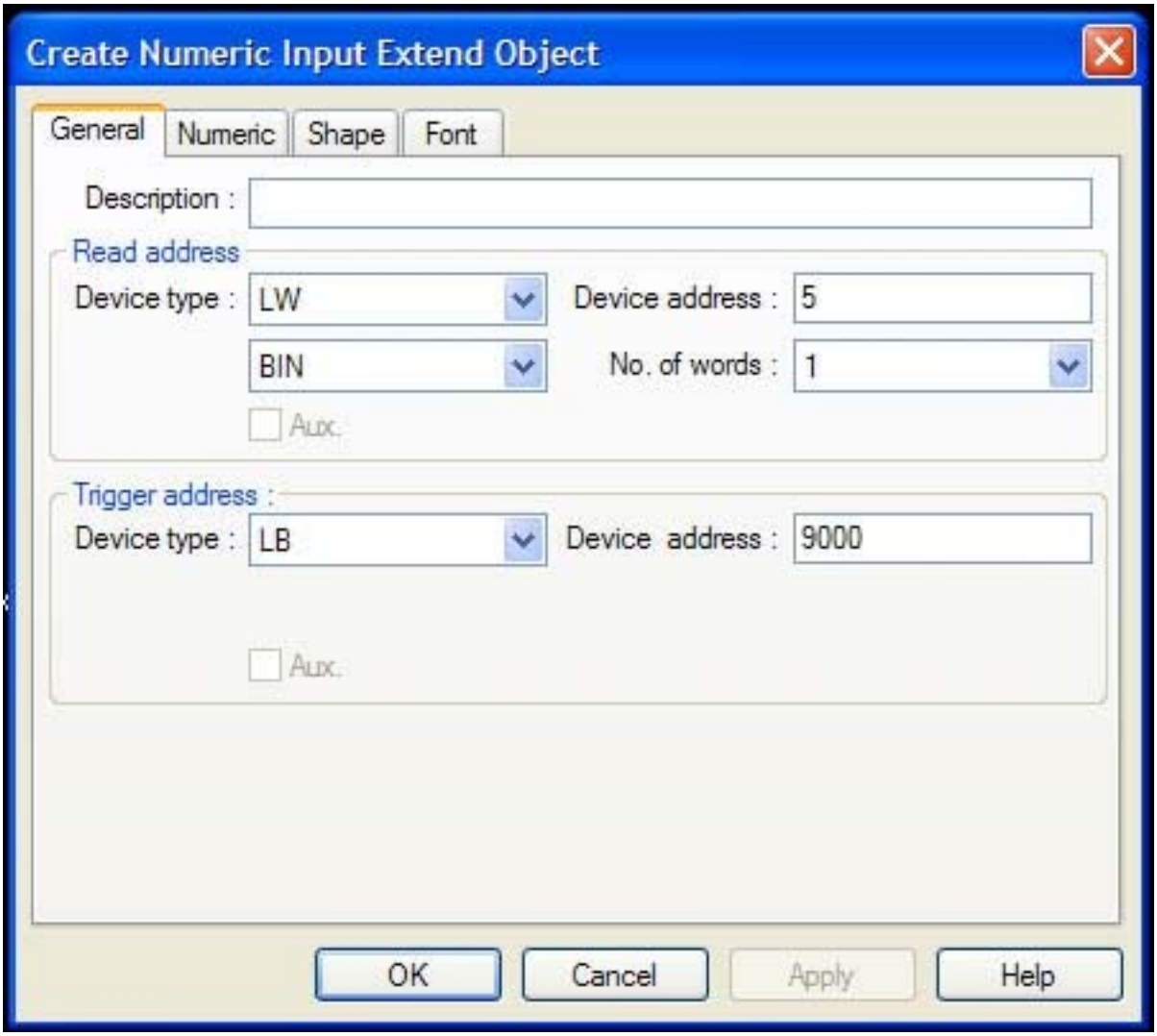
1. Start a new project file, with the initial screen configured as the startup screen. Note: this is the default setting. Display the initial screen in the work area of EasyBuilder.

2. From the **Library** menu, click **Group**, and then click **Call up Library**. The Group Library dialog box appears.



3. Click on the **Library Name:** pull-down box to select the **keypad** library.
4. Click on part 5, and then click **Place...** The Group Library dialog box closes and the keypad is placed onto the upper left corner of the work area of EasyBuilder.
5. Click on the part to move it to the lower half of the window screen.

6. From the **Part** menu, select **Numeric Input Extend**. The Create Numeric Extend Object dialog box appears.



7. Create a Numeric Input Object with the following parameters:

Tab	Frame	Parameter	Value
General	Description		Numeric Input Register
	Read Address	Device type:	LW
		Device address:	0
		BCD or BIN:	BIN
		No. of words:	1
	Trigger address	Device type:	LB
		Device address:	9000

Numeric	Display	Format	Decimal
		Type	Raw data display
	Numeric	No. above Dec:	5
		Input low:	-32768
		Input high:	32767
Shape	Shape	Use Shape	checked
		Shape library	button1, item 0
	Attribute	Color:	black
		Align:	Right Adjust
		Font:	16

For more information on how to use the Numeric Input Extend Object, consult Chapter 8 "Representing Data with Graphics Objects".

8. Create an ASCII Input Object Extend with the following parameters:

Tab	Frame	Parameter	Value
General	Description		ASCII Input Register
	Read Address	Device type:	LW
		Device address:	1
		BCD or BIN:	BIN
		No. of words:	12
	Trigger address	Device type:	LB
		Device address:	9000
	Attribute	Adjust:	Left
Shape	Shape	Use Shape	checked
		Shape library	button1, item 0
Font	Attribute	Color:	Black
		Align:	Right Adjust
		Font:	16

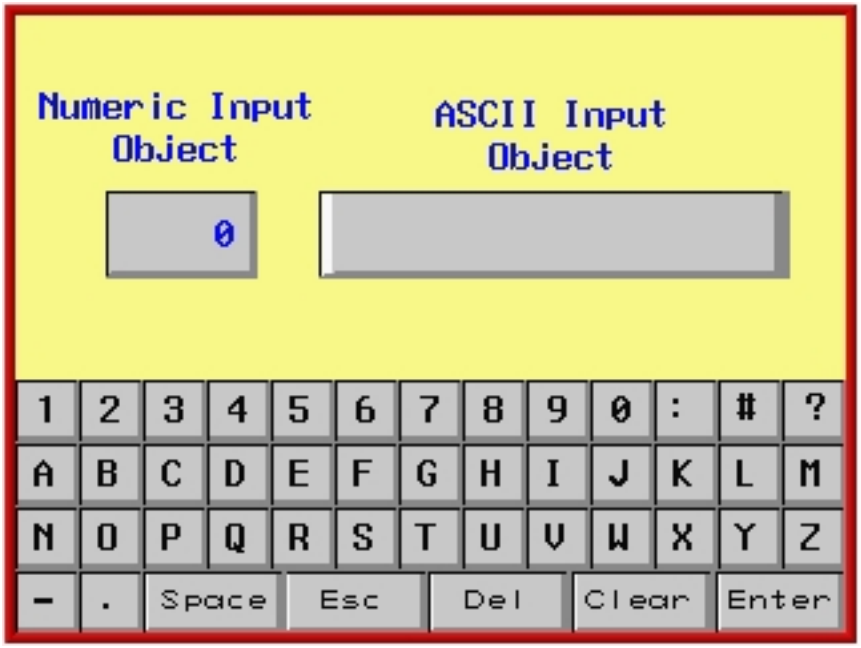
For more information on how to use the ASCII Input Extend Object, consult Chapter 8 "Representing Data with Graphics Objects".

9. Create two text boxes; one to identify the Numeric Input register, and one to label the ASCII Input register.

Editing data registers using a keypad

1. Save and compile the new project, then test the operation of the keypad by either running in Simulation mode or by downloading the project to the HMI.

2. The following screen should appear:



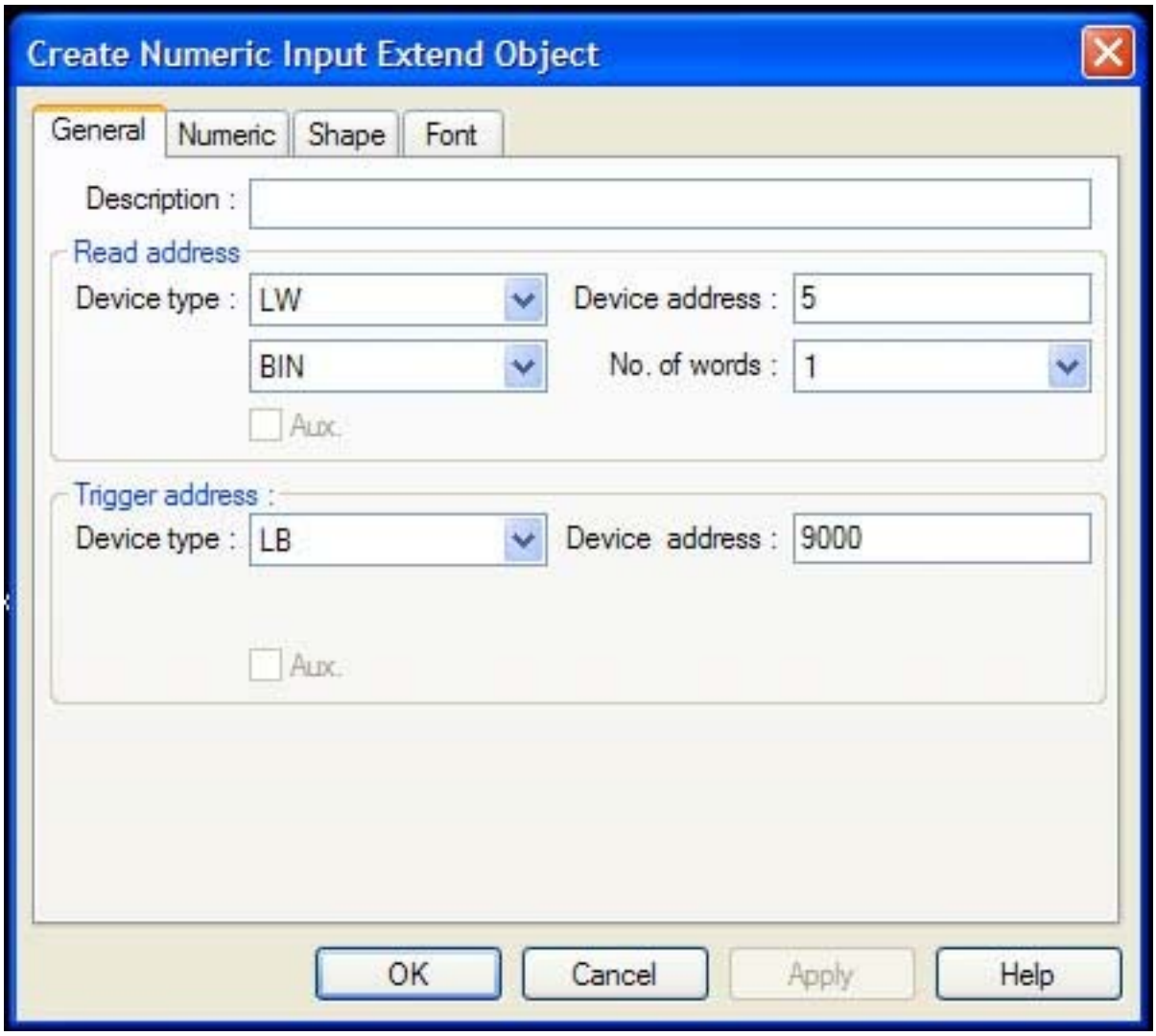
3. **Change the value of the Numeric Input register.** Touch the Numeric Input Object to activate the edit mode. This should cause the 0 digit to start blinking.
4. Using the keypad, enter digits 1, 2, 3, 4, and 5. As you enter the digits, each one should appear in the Numeric Input Object box.
5. Press the **Del** key to delete digit **5**.
6. Try pressing any of the alpha characters (A-Z). Notice that alpha characters are not accepted into a Numeric Input Object register.
7. Press the **Clear** key. Notice that the entire register is cleared to **0**.
8. Enter digits 5, 4, 3, 2, 1, and then press the **Enter** key. Notice that the last digit is still blinking and two short beeps occur. This indicates that the value entered is out-of-range. Press the **Clear** key and enter 32767. Press the **Enter** key. Notice that this value is accepted and blinking stops.
9. Touch the Numeric Input Object register again to activate editing mode. Enter five more digits then press the **Esc** key. Notice that the entry is cancelled and the previous number 32767 is restored.
10. **Change the value of the ASCII Input register.** Touch the ASCII Input Object to activate the edit mode.
11. Using the keypad, enter the character string “HELLO THERE”. As you enter the characters, each one should appear in the ASCII Input Object box. Press the **Enter** key to send the characters to the HMI internal data register.

Using the Built-In Numeric Keypad in EasyBuilder

An alternative to using the numeric keypads from the EasyBuilder libraries is using the keypad located in Window 50 of the EasyBuilder software. There are three steps to using the built-in numeric keypad - creating the data registers, setting up the common windows, and saving, compiling and simulating the keypad.

▶ To **create the Data Registers:**

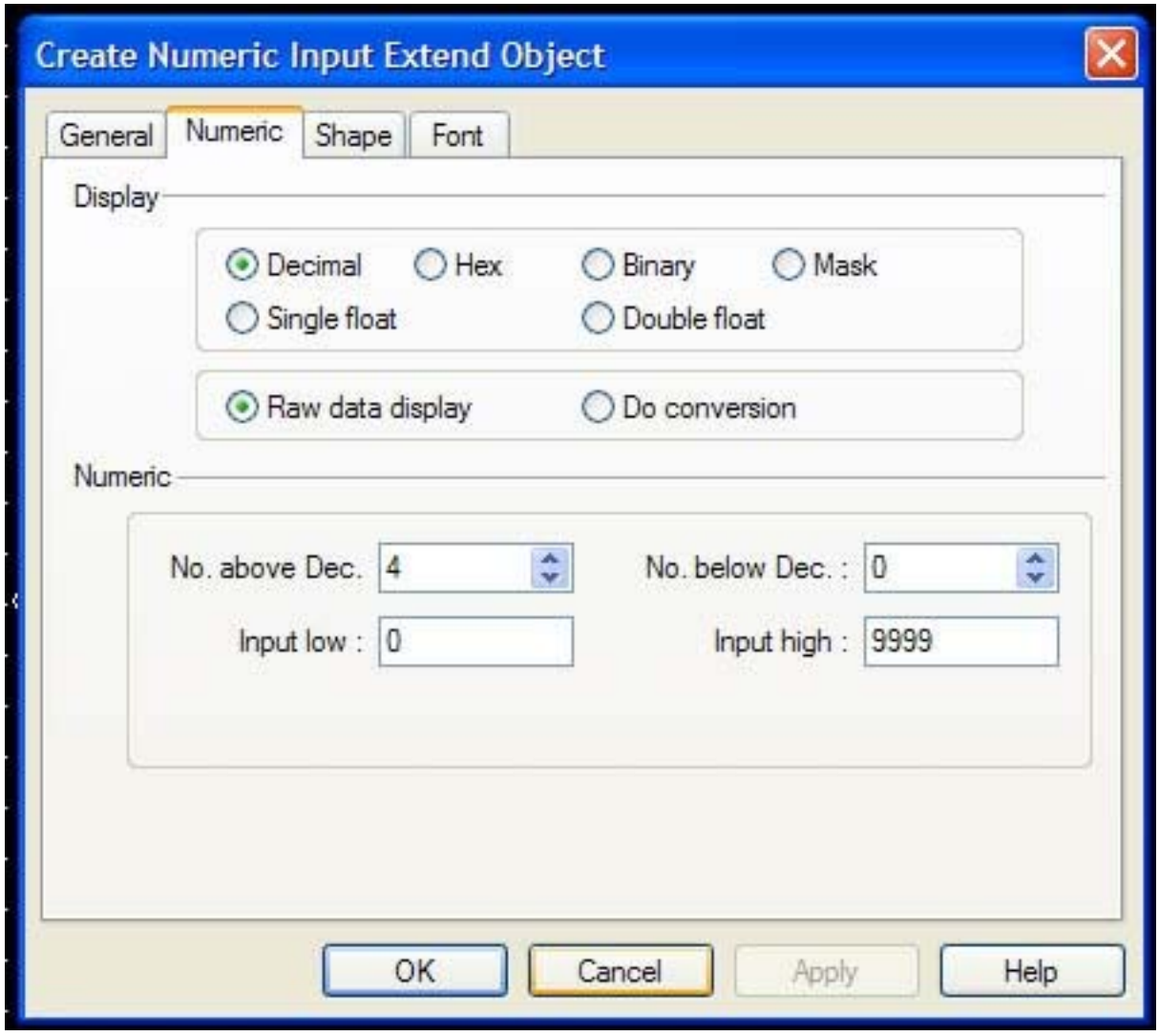
1. Select **Parts—Numeric Input Extend f**rom the menu listings. This displays the **Create Numeric Input Extend Object** dialog box.



2. Enter the following settings:

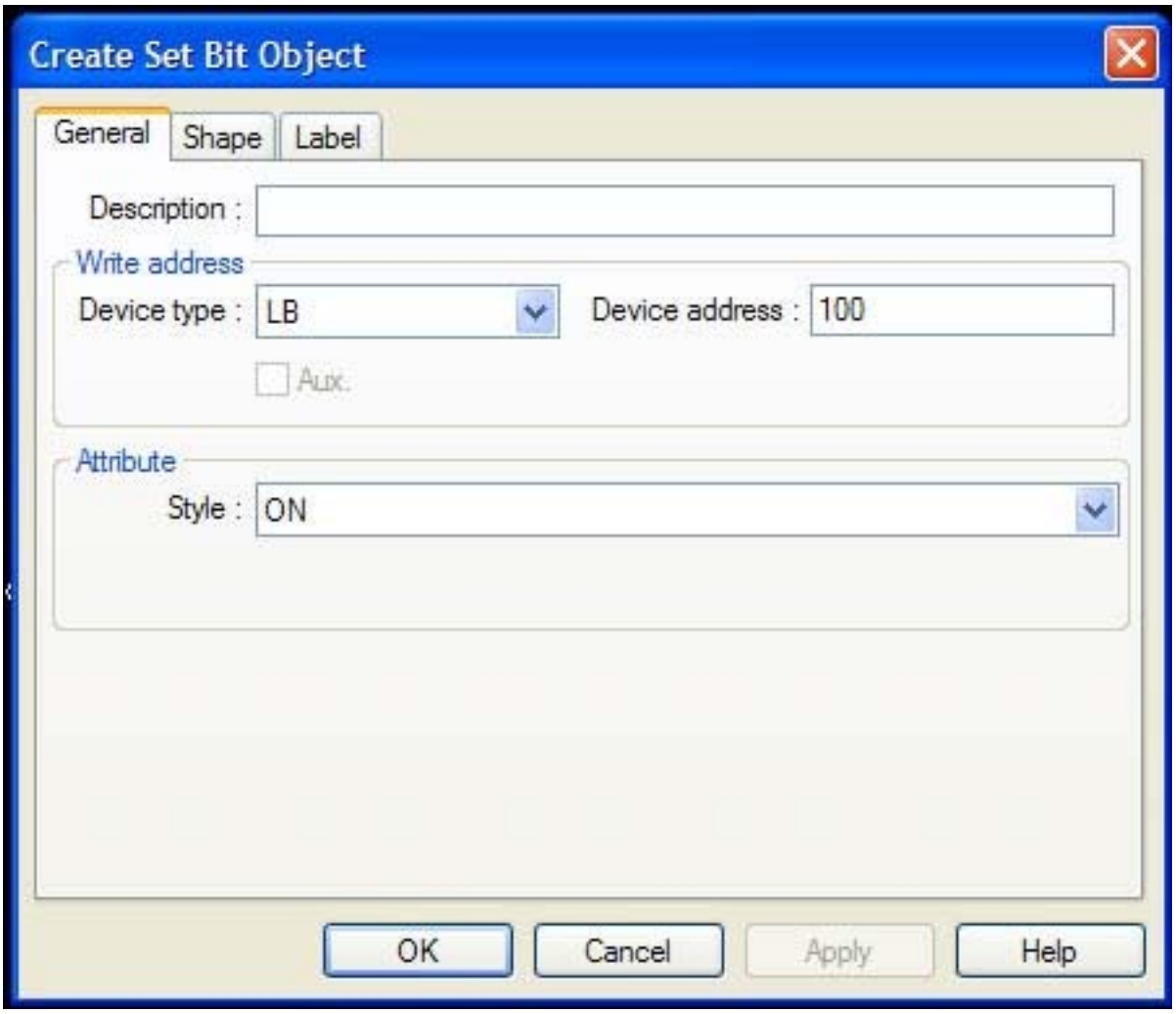
- In the Read Address: section…
  - *Device Type:* — LW
  - *Device Address:* — 5
  - *No. of Words:* — 1
- In the Trigger Address: section…
  - *Device Type:* — LB
  - Device Address: — 9000

3. Select the **Numeric** tab.



4. In the *Display* section, select the **Decimal** option button.
5. Click **OK**.
6. Use the mouse cursor to place the Numeric Input register in the center of the screen of Window #10.

7. Select Parts—Set Bit from the menu listings. This displays the **Set Bit Object** dialog box.



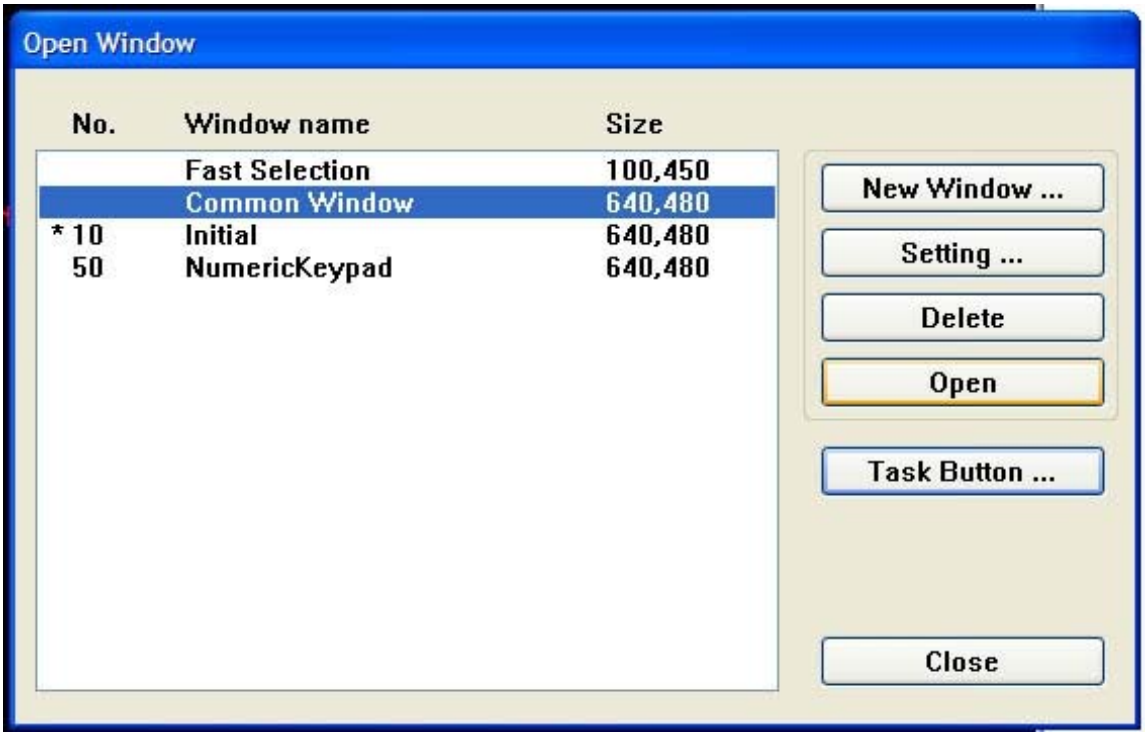
8. Enter the following settings:

- In the Write Address: section…
  - *Device Type:* — LB
  - *Device Address:* — 100
  - Attribute - Style: — ON

9. Place the Numeric Input Object and the Set Bit Object on top of each other.

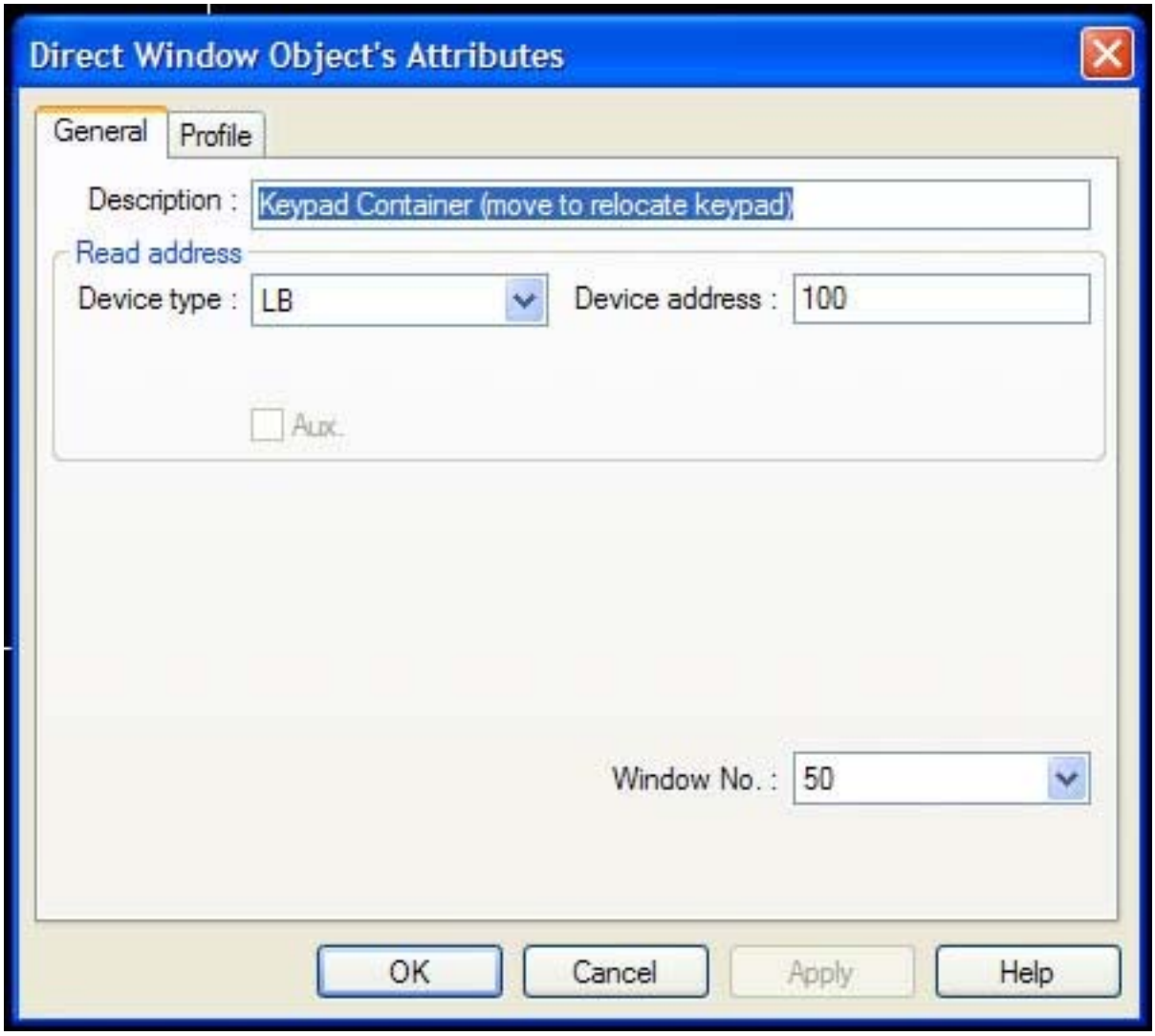
Setup the Common Window:

1. Select **Window—Open Window** from the menu listings. This displays the **Open Window** dialog box.



2. Highlight “Common Window” and click **OPEN**.
3. This window has already been configured to display a **Direct Window Object** shown as a

white outline of a square. Double click on the white outline to configure the settings. The Direct Window Object's Attributes dialog appears.



4. Verify the following sections:

- In the Read Address: section…
  - *Device Type:* — LB
  - *Device Address:* — 100
  - *Window No.:* — 50

5. Click **OK.**

To save, compile and simulate the keypad:

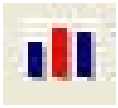
1 .Select **File—Save** from the menu listings.
2 *File name* — Name the project
3. Click **Save**
4. Select **Tools—Compile…** to compile the project.
5. Click **COMPILE**.
6. After the computer has compiled the project, click **CLOSE.**

7. Select **Tools—Offline Simulation** from the menu listings. The offline simulator with your project appears.
8. Once in Simulation mode, click on the Numeric Input object. The Decimal keypad should appear.
9. Enter **1234**. The value is entered into the register.
10. Press the **ENT** key. The Decimal keypad automatically closes.

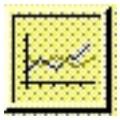
Chapter 10 - Bar Graphs, Meters, and Trends

This chapter focuses on three special graphic objects, which can be used to display PLC data registers. You read in Chapter 6 how to use shapes and bitmaps to represent the data in PLC registers as states. You also read how to use alphanumeric data fields to display the contents of PLC registers as either numbers or ASCII characters. We now introduce three more options to display the data in PLC registers:

Bar Graph Objects



Trend Display Objects



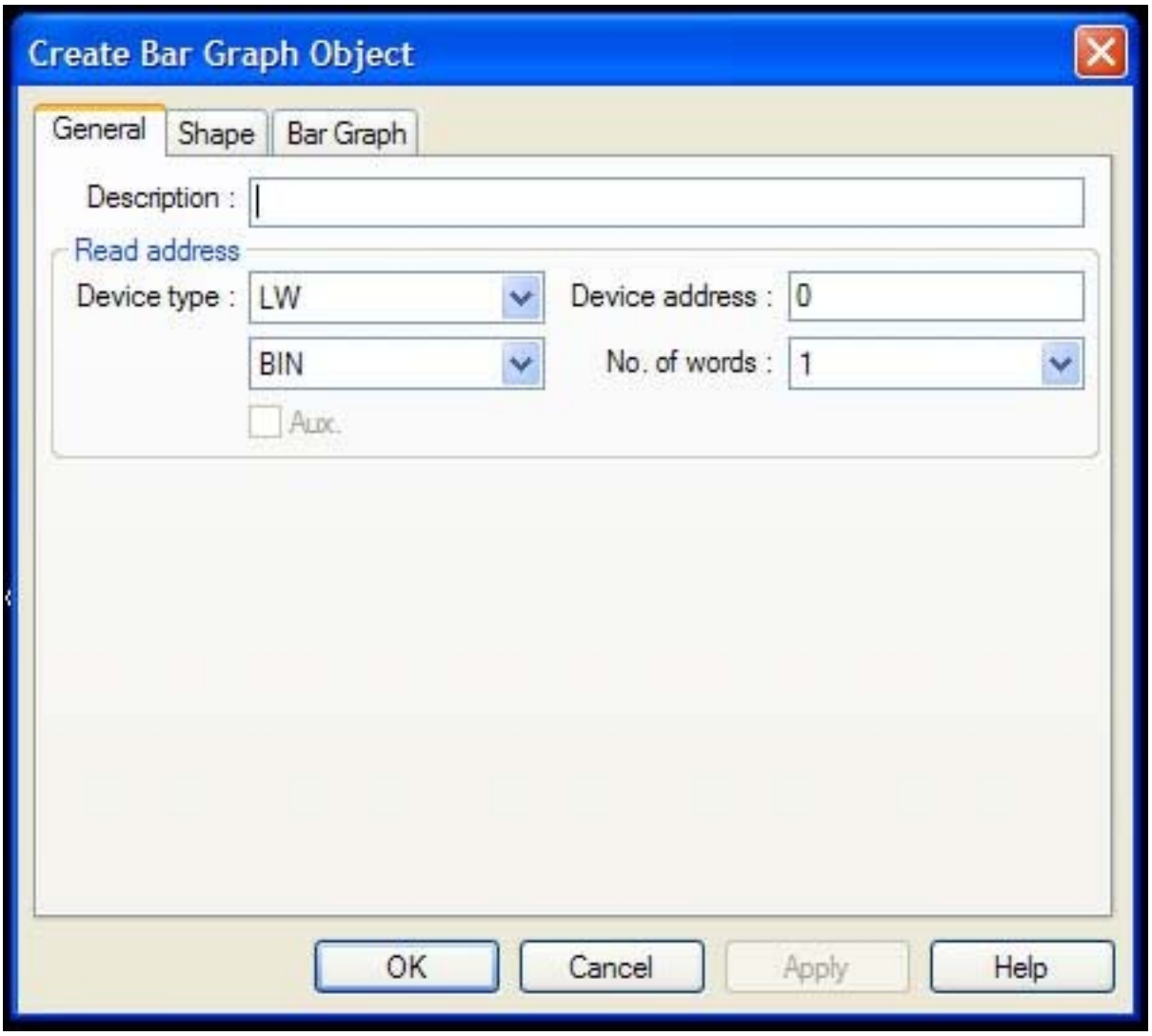
Meter Display Objects

Creating Bar Graphs

The Bar Graph Object is used to represent the data in a 16-bit PLC register as a bar graph. You can configure the bar graph to move up, down, right, or left. The bar graph can be configured with any base number that represents 0 level and any span range. You can create the bar graph with an alarm low and high setting to indicate to the HMI operator that an underflow/overflow alarm condition exists. In addition, you can set the alarm low and high limits to be controlled by two additional PLC registers so that the low and high limits are variable. Bar graphs can even be constructed with a shape or bitmap overlaying the bar graph to create flow tanks, temperature gauges, etc.

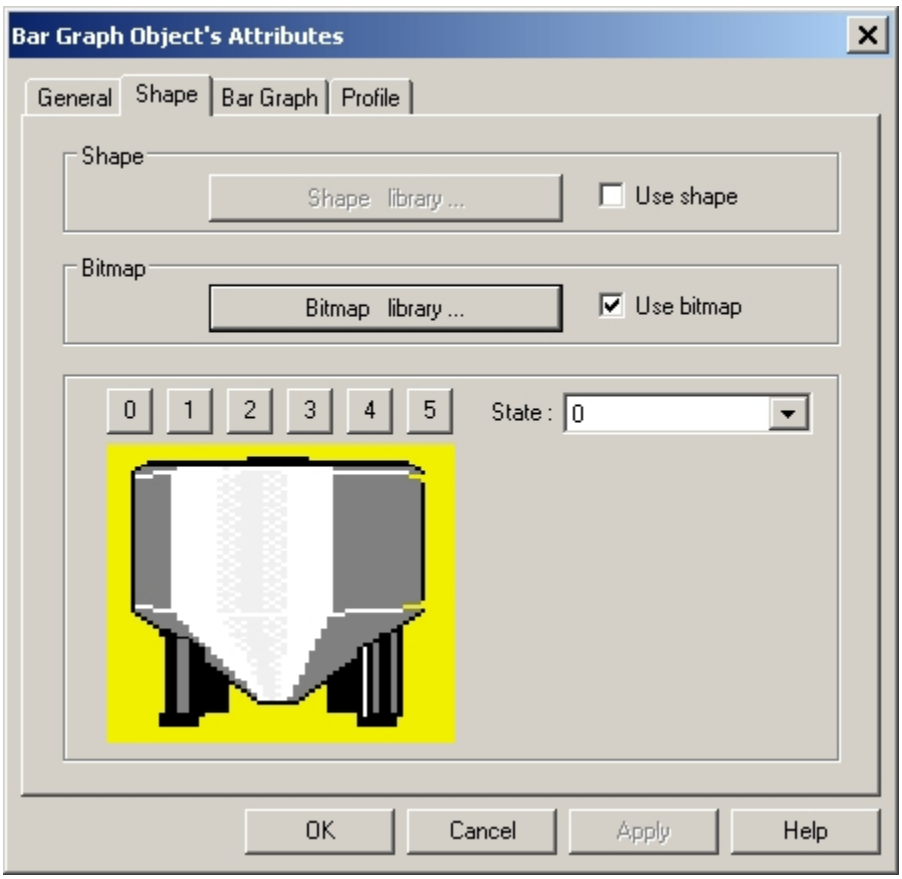
To create a Bar Graph Object

1. From the **Parts** menu, click **Bar Graph**. Or click the **Bar Graph** icon in the Part2 toolbar. The Create Bar Graph Object dialog box appears.



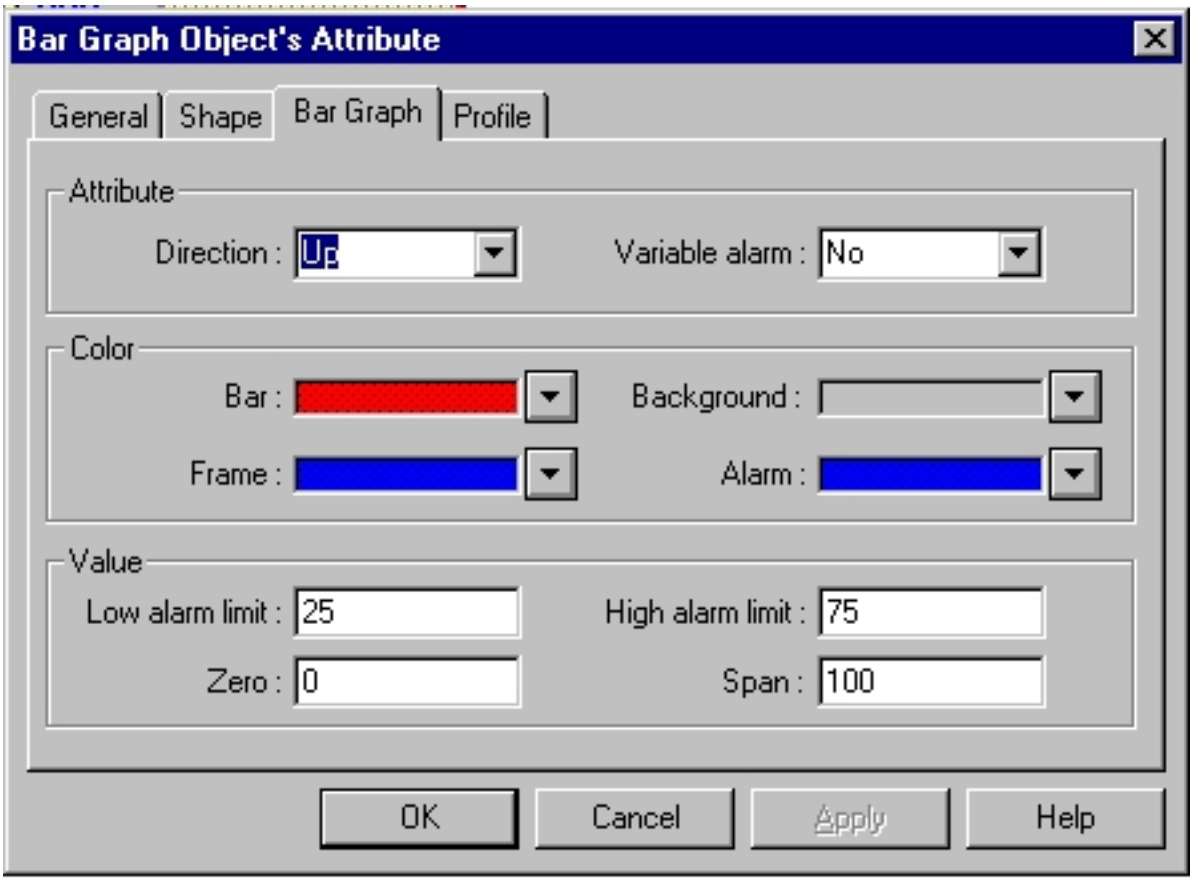
2. Use the **Description:** box to enter a title for the Bar Graph part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC register or HMI internal memory address. You can select **BIN** (binary) or **BCD** format.
4. In the **No. of words:** box, select 1 if you want a bar graph without variable alarms. Select 3 for variable alarms.

5. Click the **Shape** tab to display the Shape form.



6. A shape or bitmap is not necessary if you plan to display the bar graph as a rectangular object. If you do select a shape, the bar graph will only show in areas of the shape in which no color has been selected (i.e. inside an unfilled rectangle object). If you select a bitmap, the bar graph will only show in the area of the bitmap tagged as transparent (for example, try using the flow tank bitmaps in the PIPLINE library, items #16-37). If you need more information on how to use bitmaps and shapes, consult Chapter 6, “Creating Graphics Objects.”

7. Click the **Bar Graph** tab to display the Bar Graph form.



8. In the **Attribute** frame, select what type of Bar Graph to use:

Function	Description
Direction:	Selects which direction the bar graph should move as it increases: **Up**, **Down**, **Right**, or **Left**.
Variable alarm:	If the Variable alarm feature is selected, then the HMI continuously reads three 16-bit PLC registers instead of 1. The first register stores the current value of the bar graph. The second register stores the Low alarm limit. The third register stores the High alarm limit

9. In the **Color** frame box, select the following:

Function	Description
Bar:	Selects the color for the bar graph indicator bar.
Background:	Selects the color for the background of the bar graph.
Frame:	Selects the color for the frame. If you do not want a frame, then select the same color as used for the background color.
Alarm:	Selects the color used to indicate an alarm or out-of-range condition.

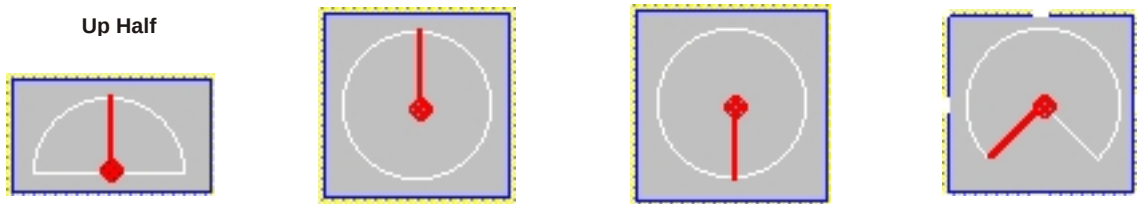
10. In the Value frame box, select the following:

Function	Range	Description
Low alarm limit:	-32768 to 32766	During operation, if the PLC register monitored by the bar graph goes below this number, the alarm color is shown for the bar graph indicator bar. If you do not want to use the alarm feature, then set this value equal to the **Zero** value. If the Variable Alarm feature is enabled, this box is deactivated since the Low alarm limit is derived from the second PLC register monitored
High alarm limit:	-32767 to 32767	Like the low limit indicator, if the PLC register monitored by the bar graph goes above this number, the alarm color is shown for the bar graph indicator bar. If you do not want to use the alarm feature, then set this value to the **Zero** value plus the **Span** value. If the Variable Alarm feature is enabled, this box is deactivated since the High alarm limit is derived from the second PLC register monitored
Zero:	-32768 to 32766	Selects the number that represents the zero level for the bar graph
Span:	-32767 to 32767	Selects the number that represents the total span of the bar graph

11. Click **OK** to close the Create Bar Graph Object dialog box. The main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
12. Once the part is placed onto the window, you can adjust the attributes of the Bar Graph by double-clicking on the part.

Creating Display Meters

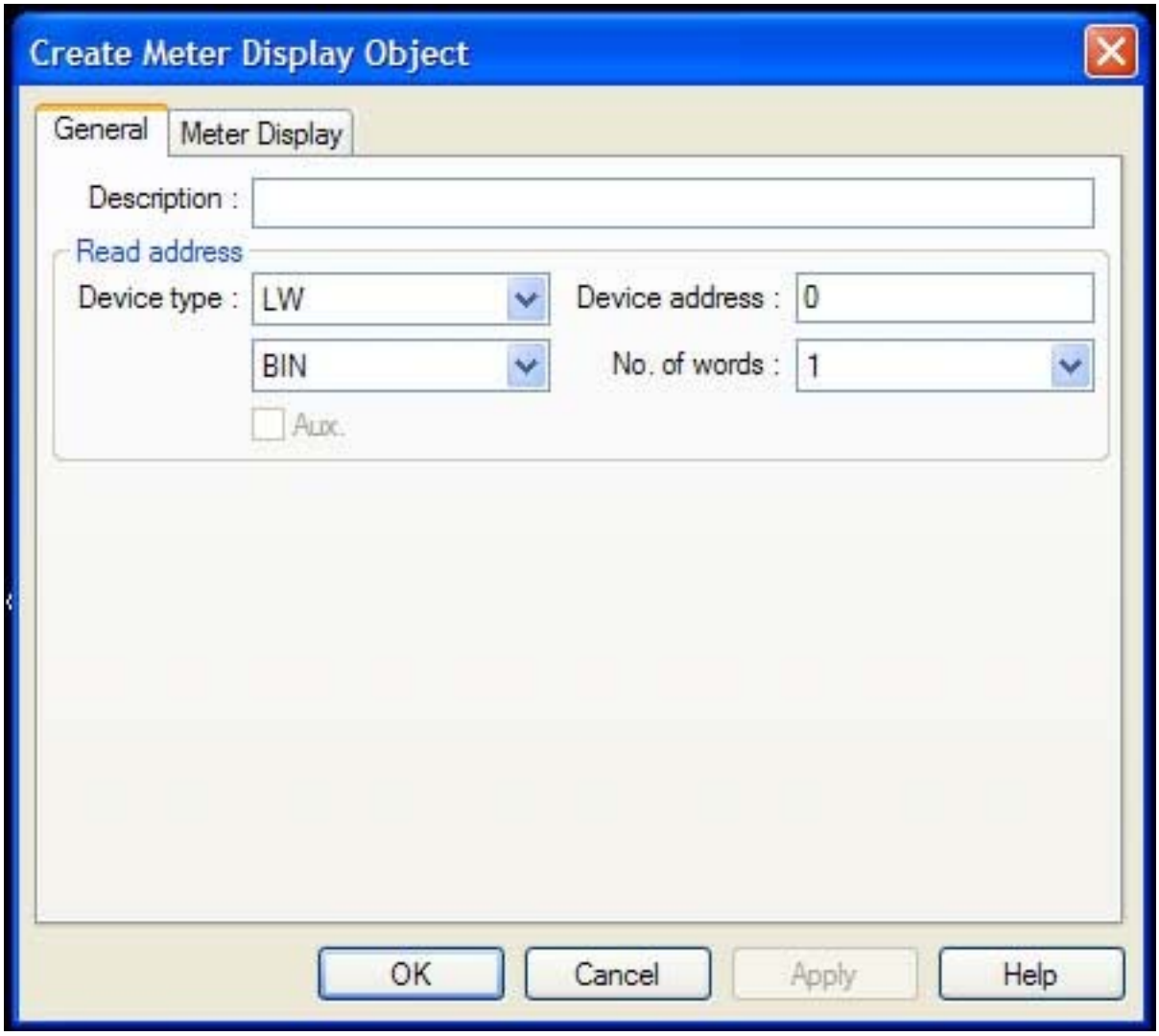
The Display Meter Object is used to represent the data in a 16-bit PLC register as a scaled meter. You can configure the Display Meter in four styles:



The Display Meter can be configured with any base number that represents 0 level and any span range. The Display Meter object is most often used with the Scale Object and a shape in the background such as a Rectangle Object.

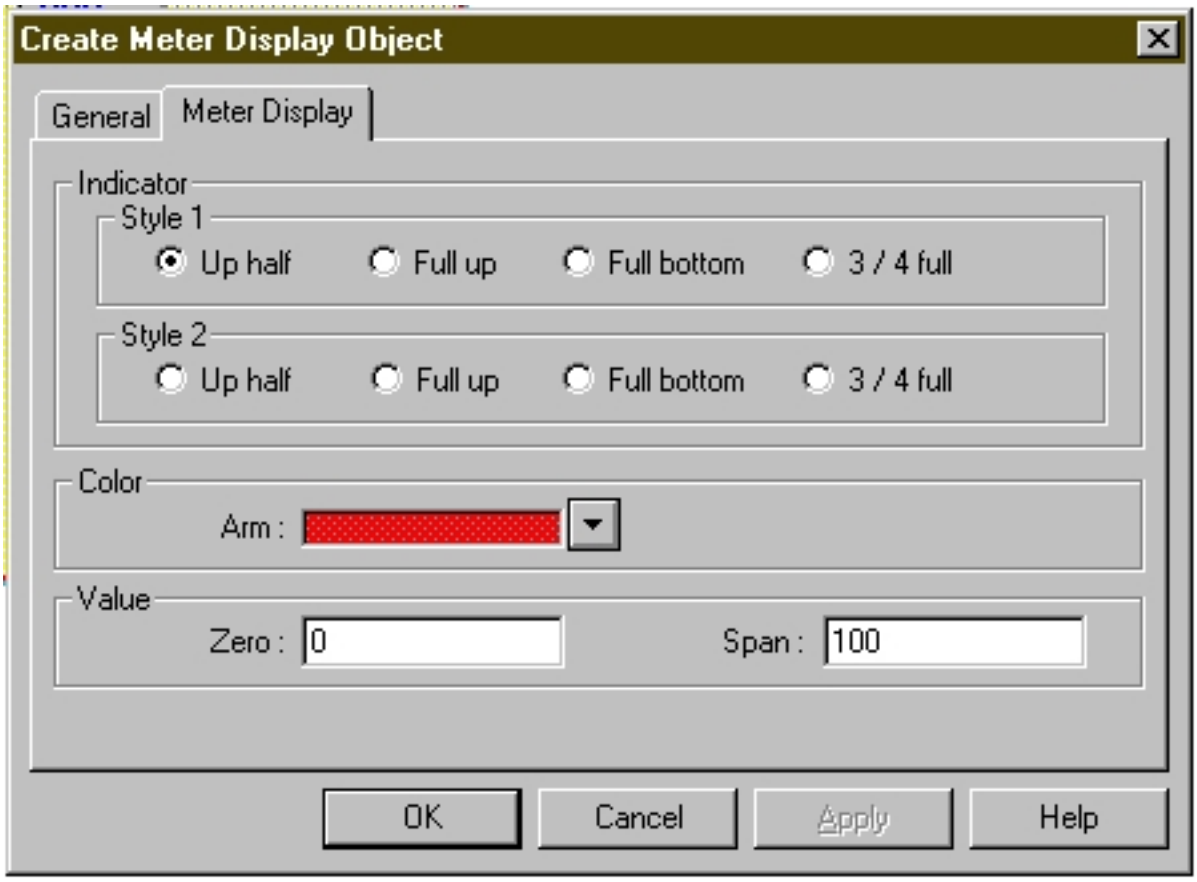
To create a Display Meter Object 

1. From the **Parts** menu, click **Meter Display**. Or click the **Meter Display** icon in the Part2 toolbar. The Create Display Meter Object dialog box appears.



2. Use the **Description:** box to enter a title for the Display Meter part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the PLC register or HMI internal memory address. You can select **BIN** (binary) or **BCD** format. In the **No. of words:** box, the only option available is 1.

4. Click the **Display Meter** tab to display the Meter Display form.



5. In the **Indicator** frame, select what type of Meter Display to use:

Style	Option	Description
Style 1 (arm with center mark)	Up half	Select for a display meter that ranges from 0 to 180 starting at the 9 o'clock position.
	Full Up	Select for a display meter that ranges from 0 to 360 starting at the 12 o'clock position.
	Full Bottom	Select for a display meter that ranges from 0 to 360 starting at the 6 o'clock position.
	¾ Full	Select for a display meter that ranges from 0 to 270 starting at the 7.5 o'clock position.
Style 2 (straight arm)	Up half	Select for a display meter that ranges from 0 to 180 starting at the 9 o'clock position.
	Full Up	Select for a display meter that ranges from 0 to 360 starting at the 12 o'clock position.
	Full Bottom	Select for a display meter that ranges from 0 to 360 starting at the 6 o'clock position.
	¾ Full	Select for a display meter that ranges from 0 to 270 starting at the 7.5 o'clock position.

6. In the **Color** frame box, select the color used for the arm or indicator of the meter.
7. In the **Value** frame box, select the following:

Function	Range	Description
Zero:	-32768 to 32766	Selects the number that represents the zero level for the meter display.
Span:	-32767 to 32767	Selects the number that represents the total span of the meter display.

8. Click **OK** to close the Create Meter Display Object dialog box. The main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.

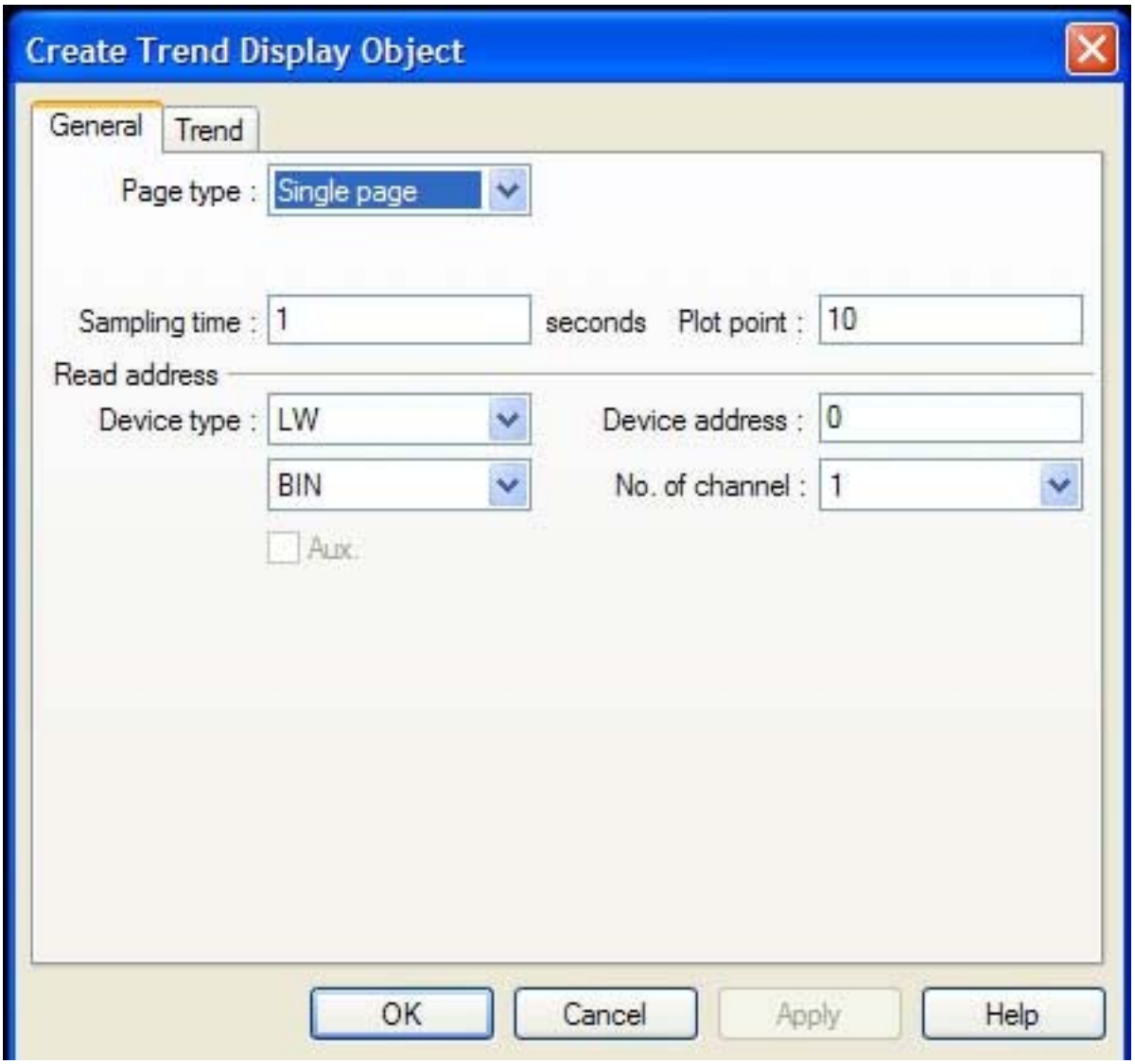
9. Once the part is placed onto the window, you can adjust the attributes of the Display Meter by double-clicking on the part.

Creating Trend Displays

The Trend Display Object is used to sample the data in a single or multiple 16-bit PLC registers and plot this data on a time graph. You can configure the Trend Display as a single page or with multiple pages that can be viewed by scrolling through the pages. The sampling rate and number of plots per page can be customized per trend graph. You can display a maximum of 16 channels with varying colors to distinguish the channels. Finally, a hold feature allows you to 'freeze' the graph and/or clear the graph.

▶ To create a Trend Display Object 

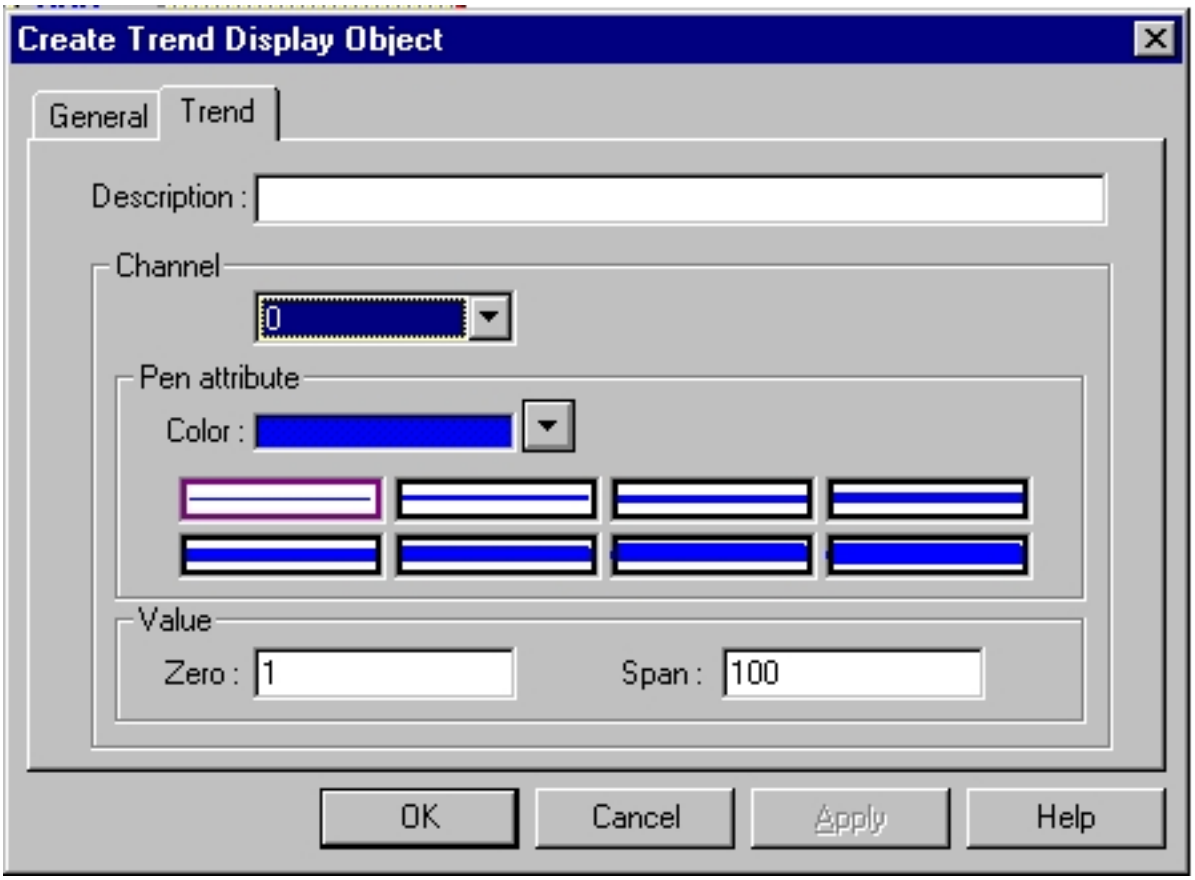
1. From the **Parts** menu, click **Trend Display**. Or click the **Trend Display** icon in the Part2 toolbar. The Create Trend Display Object dialog box appears.



2. In the **General** frame, select the following attributes:

Function	Attribute	Range	Description
Page Type:		Single page or Multiple pages	Select **Single page** or **Multiple pages**. Multiple pages configure the trend display as a 'window' to the trend graph in which different segments can be viewed by scrolling.
No. of page:		1 to 31	Available if **Multiple pages** is selected. You can select the number of pages you wish to view. Select 1 if you want to view a single page only but with the hold feature enabled.
Hold style:		Hold trend display or Hold trend display & clear	Available if **Multiple pages** is selected. Select **Hold trend display** to have the option of freezing the display. Select **Hold trend display & clear** if you want to clear the display.
Attribute:		Start from left or Start from right	Available if **Multiple pages** is selected. Option for where plotting begins.
Sampling time:		1 to 65535 (seconds)	Time interval between sampling and plotting of data
Plot point:		10 to 255	Number of times that a sampling occurs per page.
Read address:	Device type:		PLC or HMI internal memory address
	Device address:		This is the starting address that is read from to get the data to trend.
	BIN or BCD		Select binary (BIN) or BCD format
	No. of channel:	1 to 16	For each channel, the HMI reads another contiguous 16-bit PLC register
Scroll control:	Device type:		If Multiple pages is selected, this 16-bit register represents the position the window is located on the multi-page trend graph. If the value is 0, then the window is at the foremost or front of the graph. For each increment of this value, the window moves back by one plot point.
	Device address:		PLC or HMI internal memory address
	BIN or BCD		Select binary (BIN) or BCD format
Hold control:	Device type:		If **Multiple pages** is selected, this coil register is used to enable the hold function. If the coil is set, then the hold function is activated.
	Device address:		PLC or HMI internal memory address

3. Click the **Trend Display** tab to display the Trend Display form.



4. Use the **Description:** box to enter a title for the Trend Display part. A description is not necessary but does help you identify the purpose of the part
5. In the **Channel** pull-down box, select each channel to modify the attributes for the channel:

Function	Range	Description
Color:		Selects the color used for that channel.
Zero:	0	This value is always set to 0. It represents the lowest point on a trend graph for each channel.
Span:	1 to 32767	Selects the range for that channel.

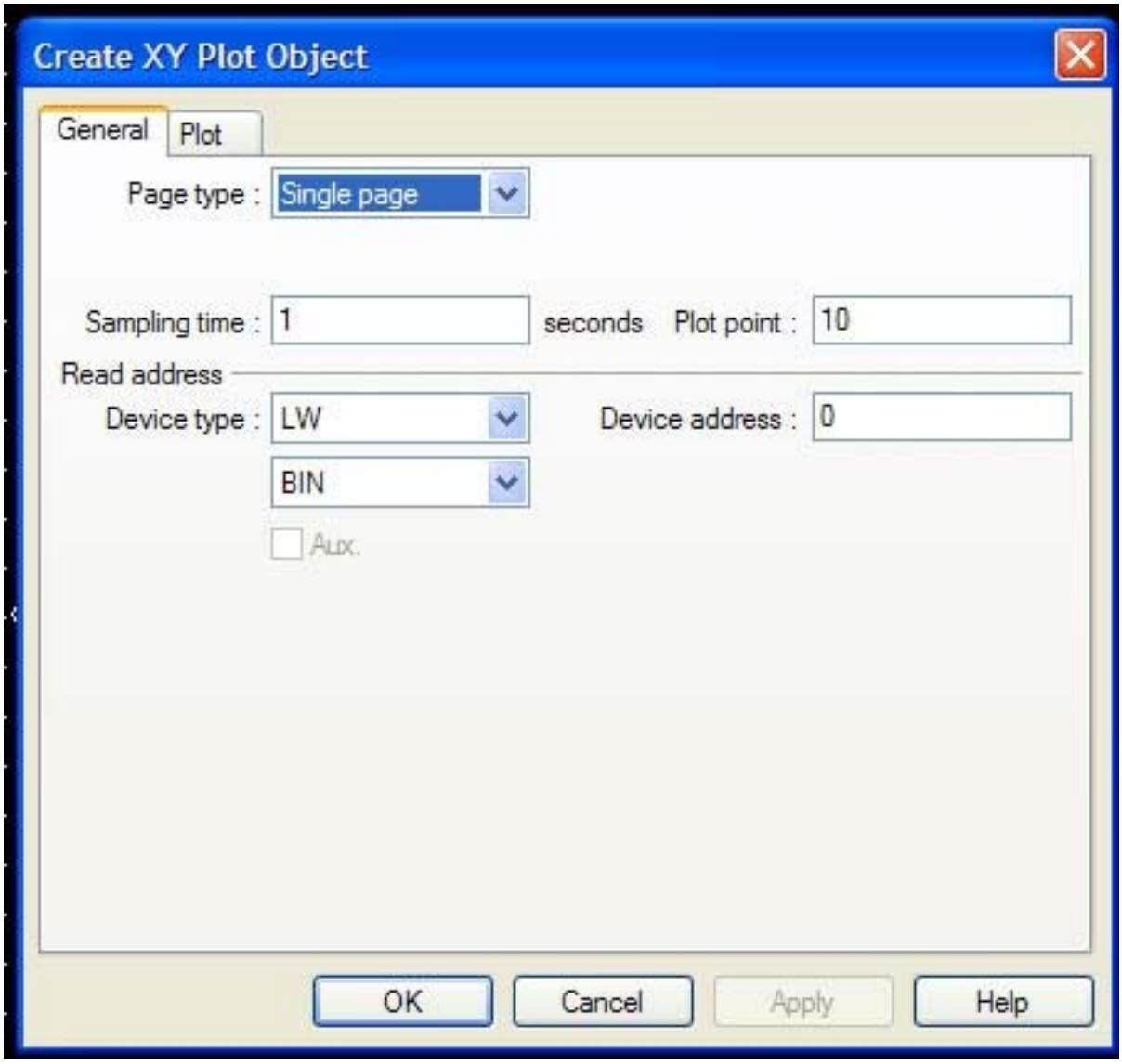
6. Click **OK** to close the Create Trend Display Object dialog box. The main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window. You may want to create a Rectangle Object that is displayed behind the Trend Graph if the Trend Display is shown in a window screen with other objects.
7. Once the part is placed onto the window, you can adjust the attributes of the Trend Display by double-clicking on the part.

Creating XY Plots

The XY Plot Object is used to sample the data in two consecutive 16-bit PLC registers and plot one register against the other. You can configure the XY Plot as a single page or with multiple pages that can be viewed by scrolling through the pages. The sampling rate and number of plots per page can be customized per plot. A hold feature allows you to 'freeze' the graph and/or clear the graph.

To create an XY Plot Object

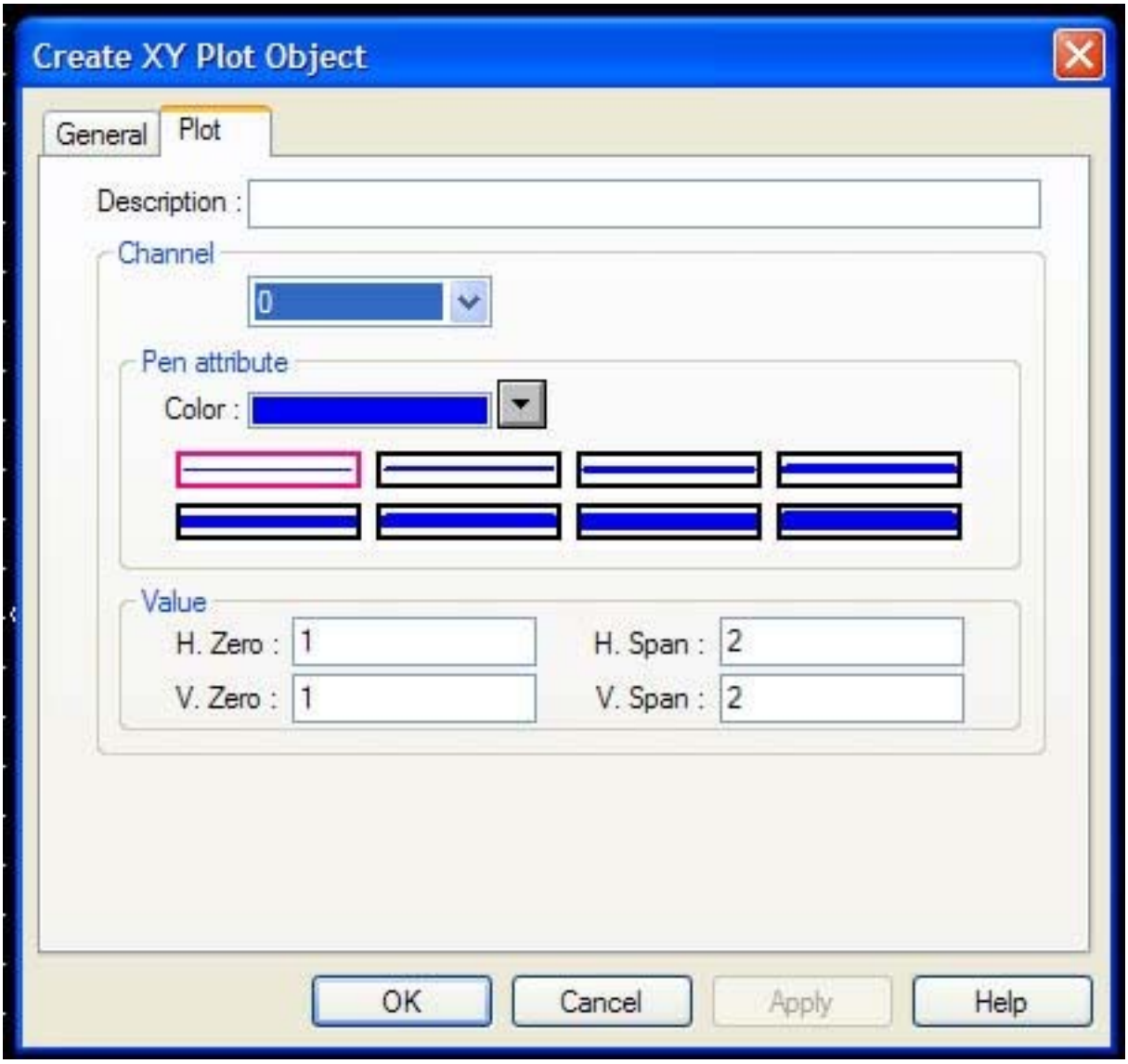
1. From the **Parts** menu, click **XY Plot**. Or click the **XY Plot** icon in the Part2 toolbar. The Create XY Plot Object dialog box appears.



2. In the **General** frame, select the following attributes:

Function	Attribute	Range	Description
Page Type:		Single page or Multiple pages	Select **Single page** or **Multiple pages**. Multiple pages configure the trend display as a 'window' to the trend graph in which different segments can be viewed by scrolling.
No. of page:		1 to 31	Available if **Multiple pages** is selected. You can select the number of pages you wish to view. Select 1 if you want to view a single page only but with the hold feature enabled.
Hold style:		Hold plot display or Hold plot display & clear	Available if **Multiple pages** is selected. Select **Hold plot display** to have the option of freezing the display. Select **Hold plot display & clear** if you want to clear the display.
Sampling time:		1 to 65535 (seconds)	Time interval between sampling and plotting of data
Plot point:		10 to 255	Number of times that a sampling occurs per page.
Read address:	Device type:		PLC or HMI internal memory address
	Device address:		This is the starting address that is read from to get the data to plot.
	BIN or BCD		Select binary (BIN) or BCD format
Scroll control:	Device type:		If Multiple pages is selected, this 16-bit register represents the position the window is located on the multi-page graph. If the value is 0, then the window is at the foremost or front of the graph. For each increment of this value, the window moves back by one plot point.
	Device address:		PLC or HMI internal memory address
	BIN or BCD		Select binary (BIN) or BCD format
Hold control:	Device type:		If **Multiple pages** is selected, this coil register is used to enable the hold function. If the coil is set, then the hold function is activated.
	Device address:		PLC or HMI internal memory address

3. Click the **Plot** tab to display the Plot Display form.



4. Use the **Description:** box to enter a title for the XY Plot part. A description is not necessary but does help you identify the purpose of the part.
5. In the **Channel** pull-down box, select each channel to modify the attributes for the channel. The XY Plot only supports channel 0.

Function	Range	Description
Color:		Selects the color used for that channel.
H. Zero:	1 to 32767	The value at which the plot will reach the left side of the Horizontal Axis.
H. Span:	1 to 32767	The value at which the plot will reach the right side of the Horizontal Axis.
V. Zero:	1 to 32767	The value at which the plot will reach the bottom of the Vertical Axis.
V. Span:	1 to 32767	The value at which the plot will reach the top of the Vertical Axis.

6. Click **OK** to close the Create XY Plot Object dialog box. The main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location. You may want to

Rectangle Object that is displayed behind the XY Plot if the XY Plot is shown in a window screen with other objects.

7. Once the part is placed onto the window, you can adjust the attributes of the XY Plot by double-clicking on the part.

Chapter 11 - Capturing Alarms and Events

This chapter looks at how the MT5xx uses alarms and events.

Using Alarms

EasyBuilder has three parts that are used to perform alarm functions: the Alarm Scan object, the Alarm Display object, and the Alarm Bar object. The Alarm Scan object monitors alarm conditions and alerts the Alarm Display object and the Alarm Bar object when an alarm condition occurs. Use the Alarm Display object if you want to display a scrollable list of all alarms that are active. Use the Alarm Bar object to display all active alarms on a single horizontally scrolling line (like a marquee).

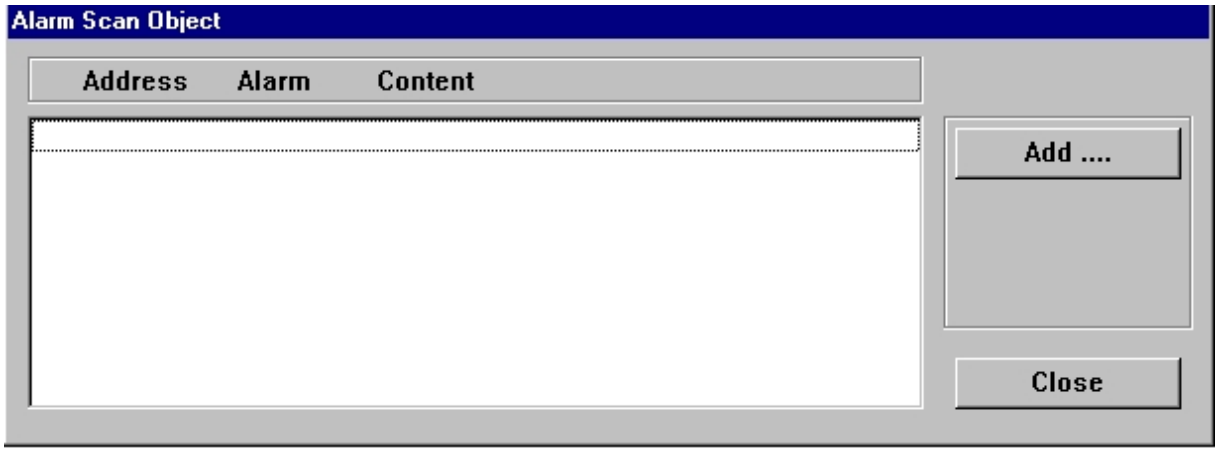
You can also use the Alarm Indicator on the Task Bar to determine if any alarms are active. For more information on using the Alarm Indicator, refer to the section

Monitoring Alarms with the Alarm Scan Object

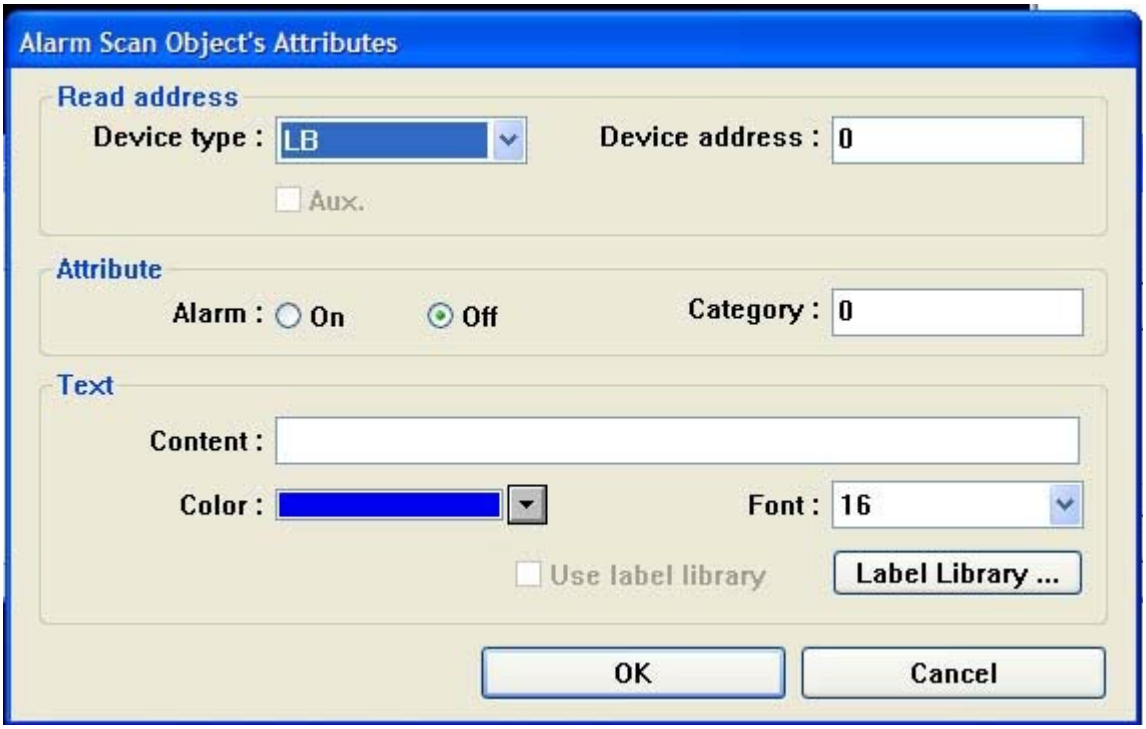
The Alarm Scan Object continuously monitors PLC coils to determine if an alarm condition has occurred. Once an alarm is active, the Alarm Scan Object directs a string of characters associated with that alarm to the Alarm Display Object and the Alarm Bar Object for display on the HMI screen.

To create an Alarm Scan Object

1. From the **Parts** menu, click **Alarm Scan**. Or click the **Alarm Scan** icon in the Draw toolbar. The Alarm Scan Object dialog box appears.



2. Click the **Add...** button. The Alarm Scan Object's Attributes dialog box appears.



3. In the **Read address** frame, select the PLC coil to monitor the alarm condition.
4. In the **Attribute** frame, for the **Alarm:** setting click **On** or **Off**. Use the On setting to activate the alarm condition when the PLC coil is set. Use the Off setting to activate the alarm condition when the PLC coil is clear. The **Category:** setting is reserved for later use.
5. In the **Text** frame, enter a text string in the **Condition:** text box that is to be displayed when an alarm condition occurs.
6. Select the color of the text string in the **Color:** box.
7. Select the font size of the text string in the **Font:** box. Sizes 16 and 24 are available.
8. Check the **Use label library** box if you would like to use the database of text labels. Click the **Label Library...**box to create a new label, if necessary.

For more information about how to use the Label Library, consult Chapter 7, ***Creating and Using Databases and Languages.***

9. Click **OK** to close the Alarm Scan Object's Attributes dialog box. The Alarm Scan Object dialog box reappears with the new entry listed. You can edit the attributes of an existing alarm by clicking on the **Setting...** button. Any entries can be deleted by clicking on the **Delete** button.
10. Click **Close** to close the Alarm Scan Object dialog box. The Alarm Scan Object is active no matter what windows are currently displayed on the HMI screen. Therefore, you do not have to place the object onto a window screen.

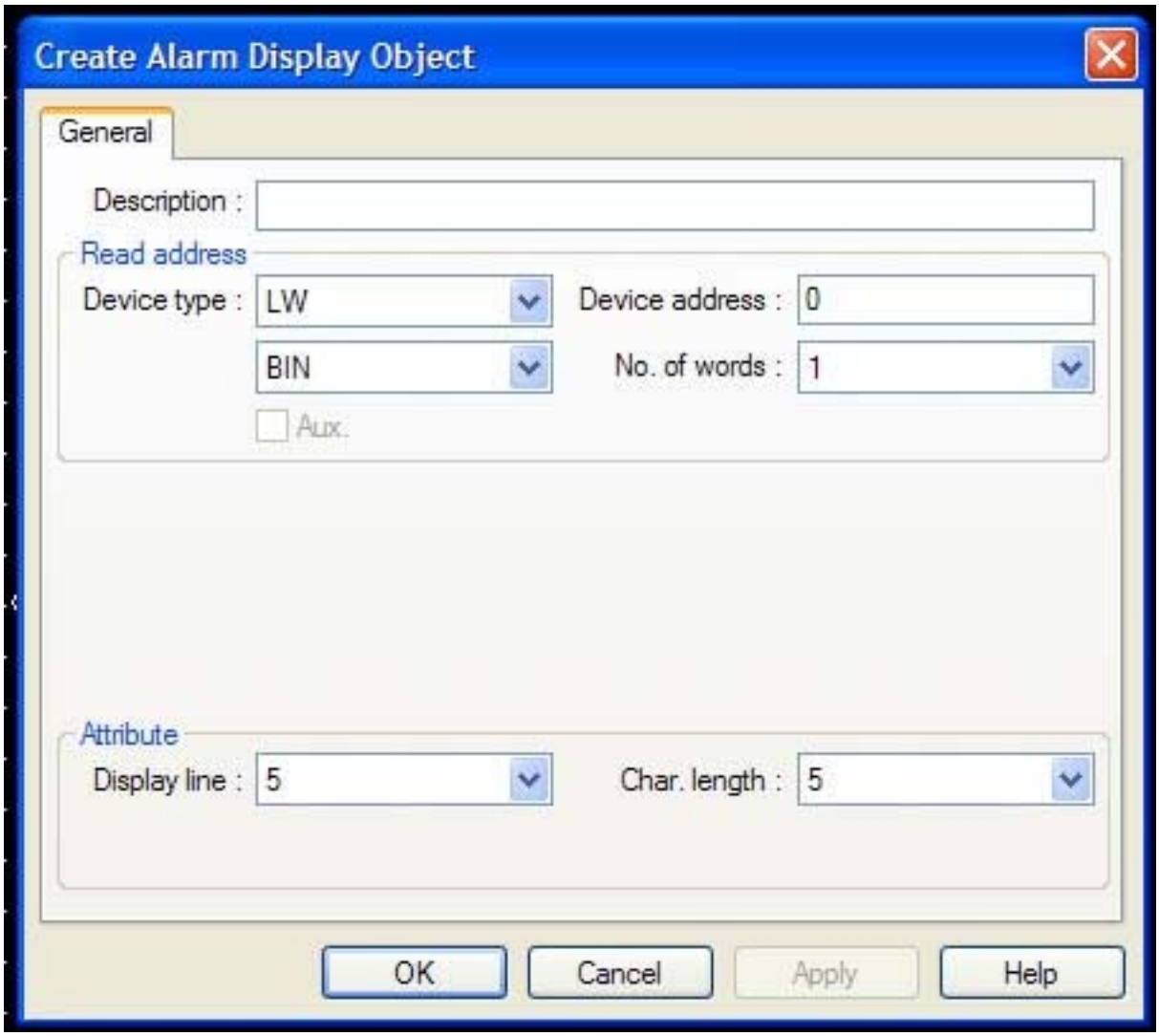
Displaying Alarms using the Alarm Display Object

Though the Alarm Scan Object continuously monitors the PLC for alarms, it cannot display the alarms without the Alarm Display Object or the Alarm Bar Object. The Alarm Display Object takes the alarm strings sent to it from the Alarm Scan Object and displays them on a window screen as a list. The list contains all of the active alarms occurring with the most recently activated alarm at the top of the list. You do not have to display the entire list of alarms on a window screen. You can limit the viewable alarms to a 'scrollable window' that displays a given

number of lines. You can then create a register that allows you to scroll through the list of alarms using the scrollable window.

To create an Alarm Display Object

1. From the **Parts** menu, click **Alarm Display**. Or click the **Alarm Display** icon in the Part2 toolbar. The Create Alarm Display Object dialog box appears.



2. Use the **Description:** box to enter a title for the Alarm Display part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Read address** frame, select the 16-bit PLC register or HMI internal memory address that will be monitored by the Alarm Display part for the scrollable window. You can select **BIN** (binary) or **BCD** format.
4. The **No. of words:** box is set at 1 and cannot be changed.

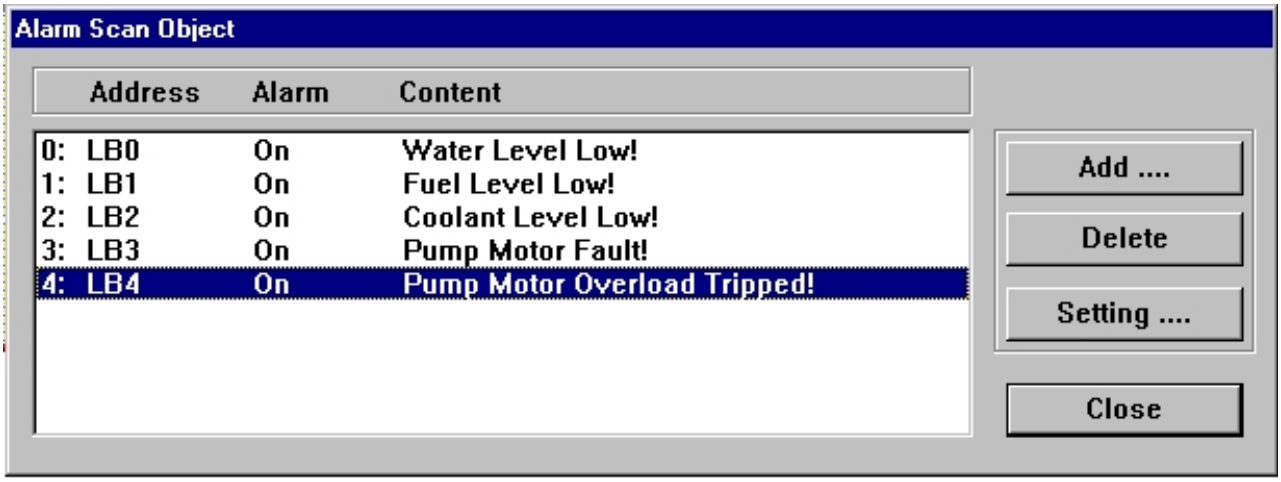
5. In the **Attribute** frame, select what type of options the Alarm Display uses:

Function	Range	Description
Display line:	1 to 14	Selects how many alarms to display in the scrollable window. The number of lines refers to lines that are Size 16 font. If the alarm text string is Size 24 font, then two lines are required for every alarm displayed.
Char length:	1 to 39	Selects how long the scrollable window is. Select a length that is equal to the maximum length of the largest alarm text string. If Size 24 font is used for the alarm text string, then two characters are required for each character in the text string.

6. Click **OK** to close the Create Alarm Display Object dialog box. The main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
7. Once the part is placed onto the window, you can adjust the attributes of the Alarm Display by double-clicking on the part.

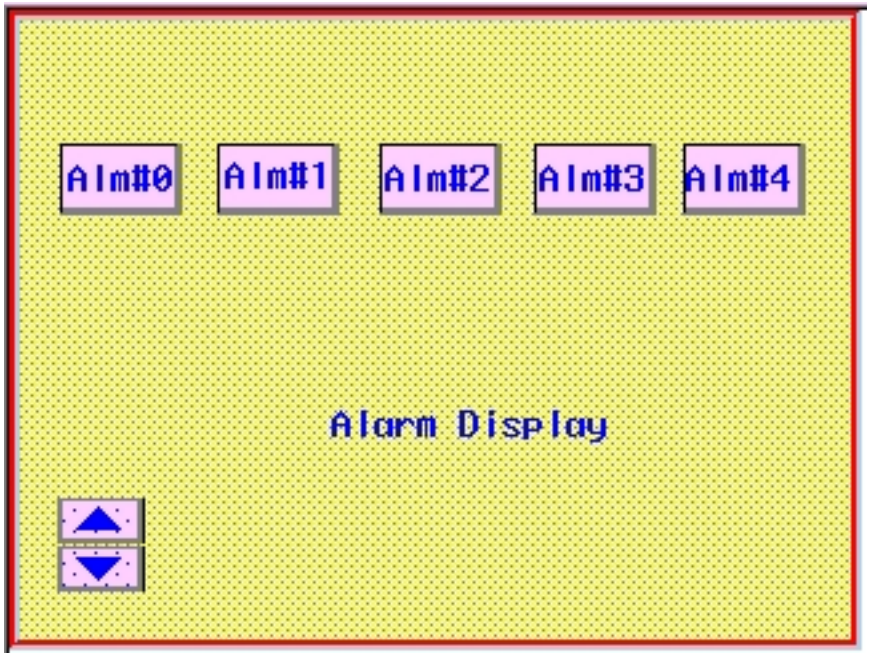
Using the scrollable window feature

Let's go through an example to illustrate how the Alarm Display Object is used with the scrollable window. First, use the Alarm Scan Object to create five alarms:

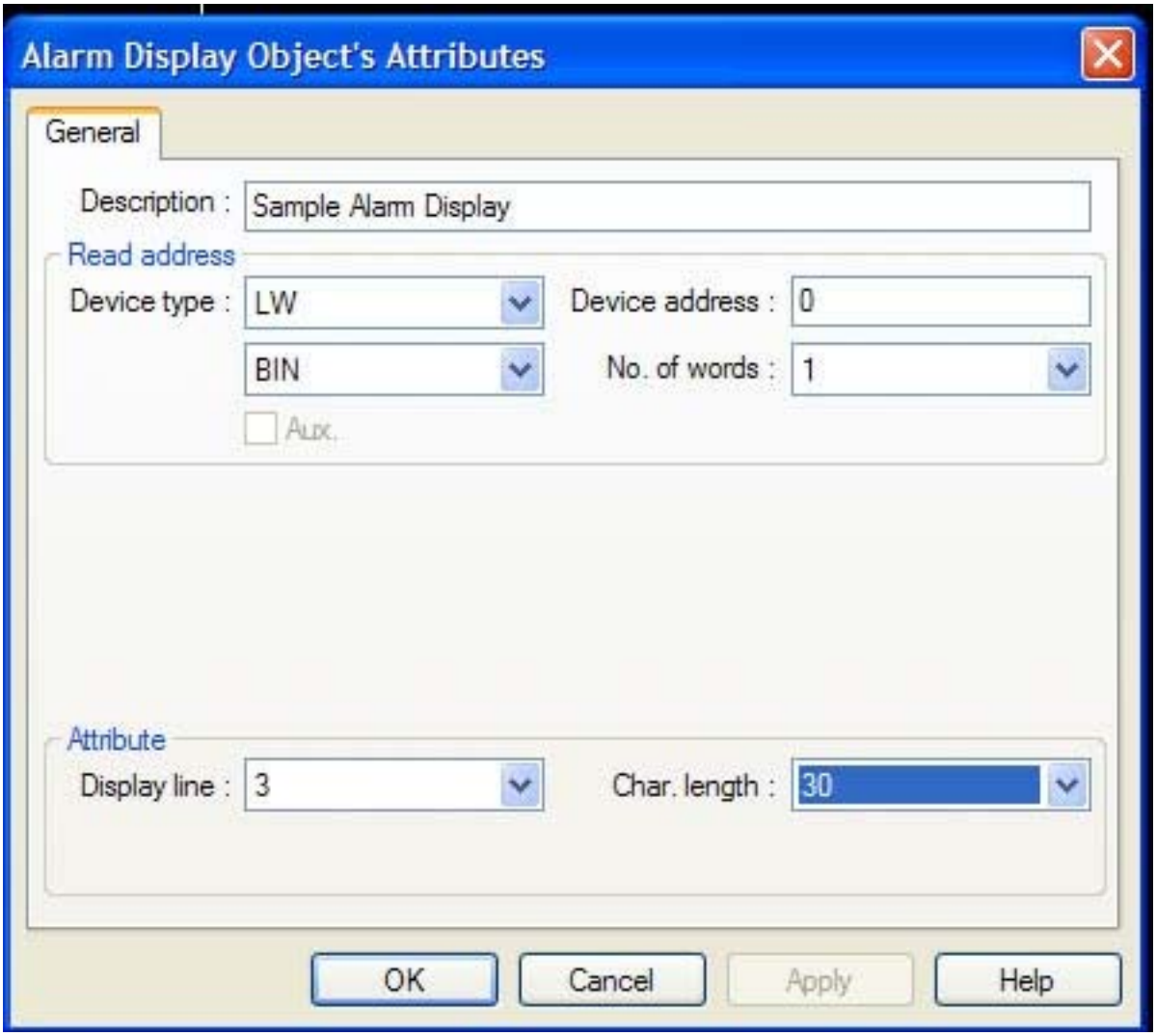


Create five Set Bit objects with the Toggle attribute that can write to HMI coils LB0-4. Place these onto a startup screen of EasyBuilder. Create a Text Object with the string "Alarm Display". Finally, create two Set Word Objects with the Add value (JOG+) attribute and Sub value (JOG-) attribute that increment/decrement by 1. Both parts should use HMI register LW0. These two objects shall be used to adjust the scrolling window of the Alarm Display Object.

When you have finished, the startup screen should look like the following:

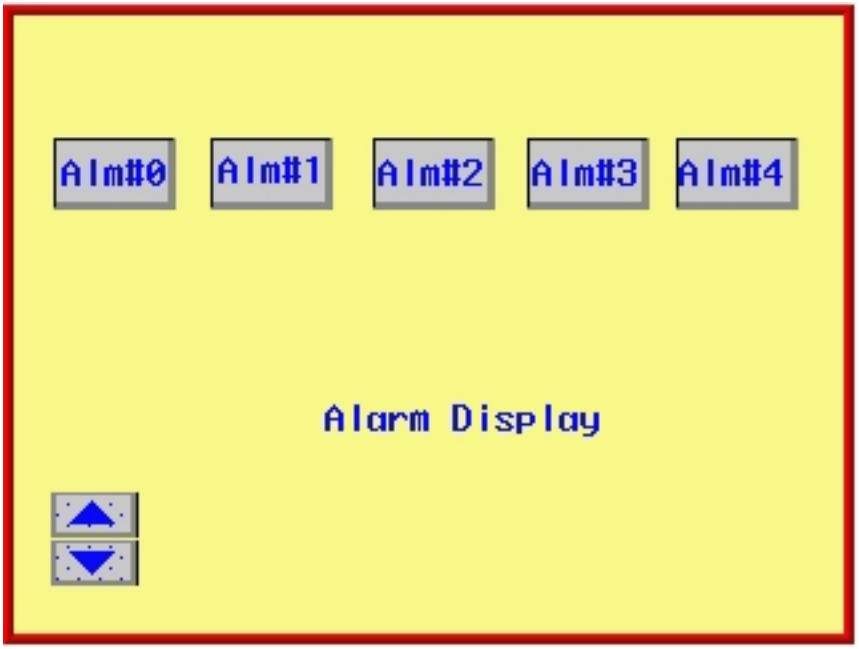


Now create an Alarm Display object with the following attributes:



Place the Alarm Display object directly underneath the “Alarm Display” text string and to the right of the two Set Word objects.

Save and compile the project, then simulate off-line. The startup screen should appear with the following displayed:



Press the Alm#0, Alm#1, and Alm#2 keys to activate three alarms. As each alarm is activated it is immediately displayed in the Alarm Display scrollable window. The most recent alarm is always displayed at the top of the list. Notice that all three alarms can be seen in the scrollable window since we configured the Alarm Display object with a scrollable window of three lines.

Now press the Alm#3 and Alm#4 keys to activate those alarms. Since the scrollable window is configured for three lines, two of the alarms on the list cannot be seen.

To see the other alarms, you can press the Down Arrow key. This will increment the value in LW0, which is used by the Alarm Display object to determine what part of the alarm list to show with the scrollable window. Use the Up Arrow key to return to the top of the list.

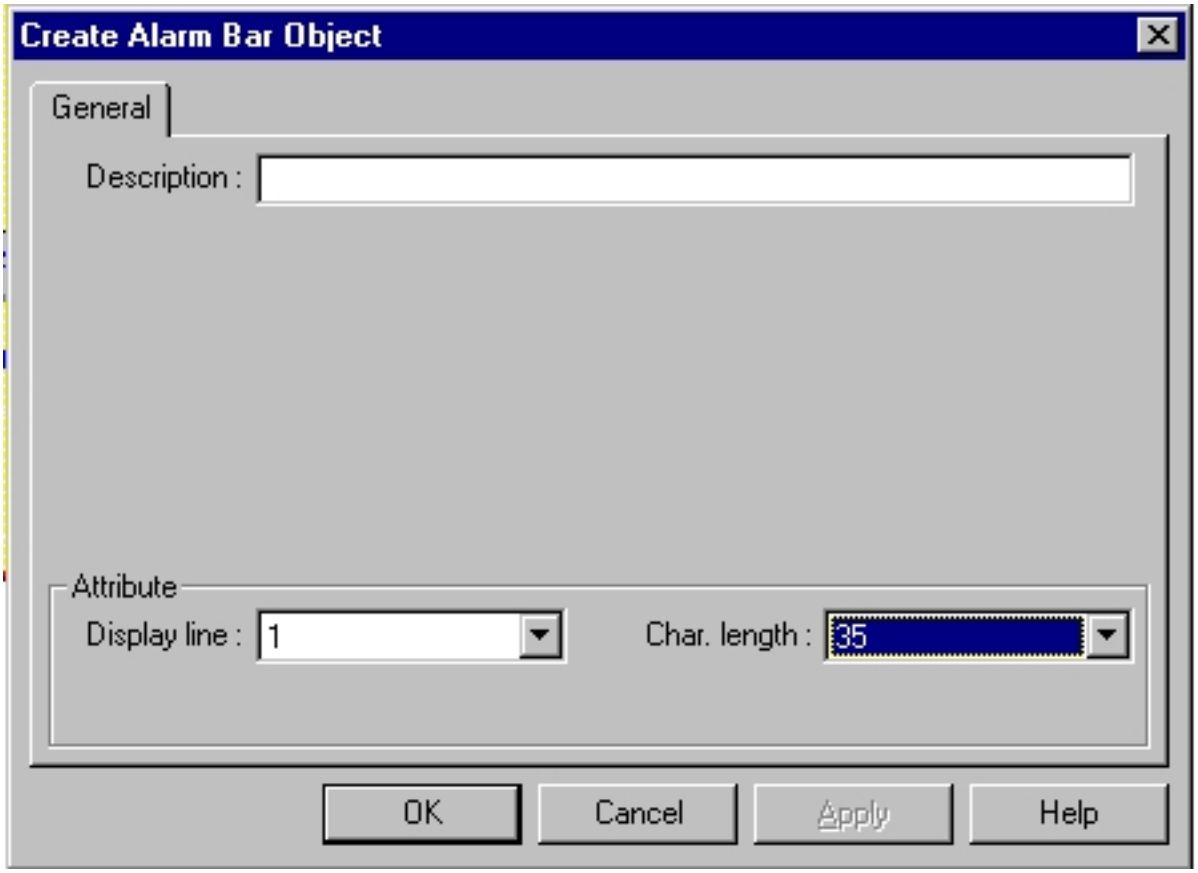
Using the scrollable window feature on the Alarm Display object allows the HMI operator to see all the currently active alarms without allocating a large area of the HMI screen to see them.

Displaying Alarms using the Alarm Bar Object

The Alarm Bar Object displays alarms scrolling horizontally along a single line. The alarm bar continuously scrolls each alarm until the alarm is no longer active. If more than one alarm is active, the Alarm Bar Object appends each alarm to the string of characters scrolled. This part can be used on window screens where space is very limited. It also allows you to display long alarm text strings that are too long to be shown in the Alarm Display Object.

To create an Alarm Bar Object 

1. From the **Parts** menu, click **Alarm Bar**. Or click the **Alarm Bar** icon in the Part2 toolbar. The Create Alarm Bar Object dialog box appears.



2. Use the **Description:** box to enter a title for the Alarm Bar part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Attribute** frame, select what type of options the Alarm Bar uses:

Function	Range	Description
Display line:	1 to 14	Selects the size of the alarm bar. Although you can select a size of 1 to 14, the Alarm Bar only displays a single line. Therefore, select 1 for alarms that use the Size 16 font, and select 2 for alarms that use the Size 24 font.
Char length:	1 to 39	Selects how long the alarm bar is. If Size 24 font is used for the alarm text string, then two characters are required for each character in the text string.

4. Click **OK** to close the Create Alarm Bar Object dialog box. The main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
5. Once the part is placed onto the window, you can adjust the attributes of the Alarm Bar by double-clicking on the part.

The Alarm Bar Object scrolls through each character of an alarm string at a predefined rate. You can change the rate at which the characters scroll by doing the following.

To change the scrolling rate

1. From the **Edit** menu, click **System Parameters**. The Set System Parameters dialog box appears.
2. Click the **General** tab. The General form is displayed.
3. In the **Alarm bar** frame, select what type of scroll options the Alarm Bar uses:

Function	Range	Description
Pixels per scroll:	8, 16, 24, 32	Selects the number of pixels that shift with each scroll.
Scroll speed:	0.1 to 25.5	Selects the time interval between shifts in the scrolling in seconds.

4. Click **OK** to close the Set System Parameters dialog box. The main screen of EasyBuilder reappears.

Using Events

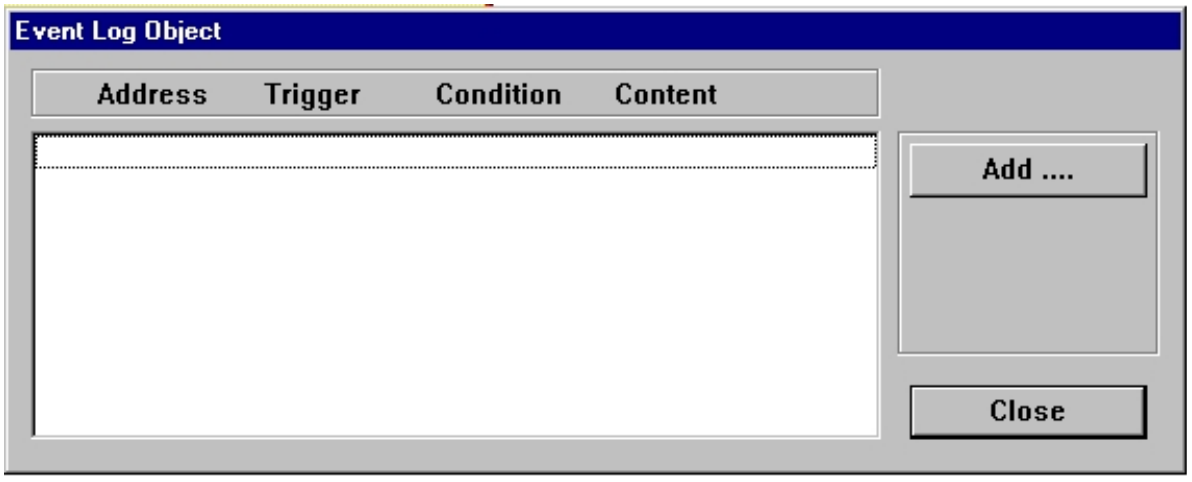
EasyBuilder has two parts that are used to perform Event functions: the Event Log object and the Event Display object. The Event Log object monitors events and alerts the Event Display when an event is triggered. The Silver Series monitors a maximum of 200 events but you can program the HMI to monitor up to an additional 1000 events!

Monitoring Events With the Event Log Object

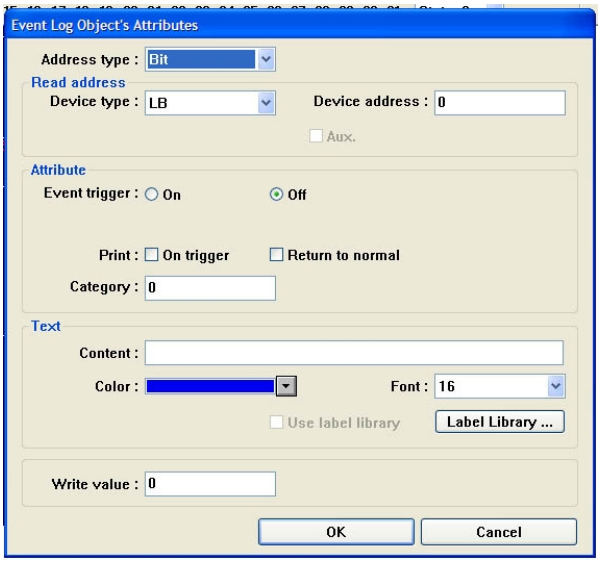
The Event Log Object continuously monitors PLC coils and registers to determine if an event has occurred. Once an event has triggered, the Event Log Object directs a string of characters associated with that event to the Event Display Object for display on the HMI screen. The HMI operator can acknowledge each event by touching the string of characters that have been displayed. Once acknowledged, the Event Log Object has the option of calling a popup window.

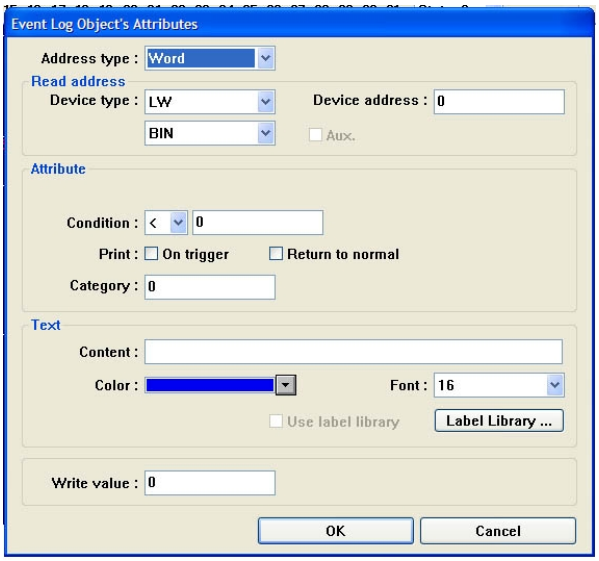
To create an Event Log Object 

1. From the **Parts** menu, click **Event Log**. Or click the **Event Log** icon in the Draw toolbar. The Event Log Object dialog box appears.



2. Click the **Add...** button. The Event Log Object's Attributes dialog box appears.





3. In the **Address type:** box, select **Bit** to trigger on the value of a PLC coil. Select **Word** to trigger on a value or range of values in a PLC 16-bit register.
4. In the **Read address** frame, select the PLC coil or register to monitor for the event. For word addresses, select binary **BIN** or **BCD** format.
5. In the **Attribute** frame, select what will cause the event to be triggered:

Function	Description
Event Trig:	For PLC coils only. Select **On** to trigger the event when the PLC coil is set. Select **Off** to trigger the event when the PLC coil is clear.
Condition:	For PLC registers only. Select the less than sign (**<**) to trigger on any value that is less than the constant entered. Select the greater than sign (**>**) to trigger on any value that is greater than the constant entered. Select the equals (==) to trigger when the value is equal to the constant entered. Then enter a constant value between -32767 and +32767
Print:	On the MT510, causes the text in the Contents box to be printed when the event is triggered or returns to normal.
Category:	Not used. Reserved for later use.

6. In the **Text** frame, enter a text string in the **Content:** text box that is to be displayed when an event is triggered.
7. Select the color of the text string in the **Color:** box.
8. Select the font size of the text string in the **Font:** box. Sizes 16 and 24 are available.
9. The **Write Value:** text box can be used to display a popup window on the HMI screen as a result of acknowledging a triggered event. Enter the number of the window to be displayed. If 0 is entered, this option is disabled.
10. Check the **Use label library box** if you would like to use the database of text labels. Click the **Label Library...** box to create a new label if necessary.

*For more information on how to use the label library, consult **Chapter 7, Creating and Using Databases and Languages***

11. Click **OK** to close the Event Log Object's Attributes dialog box. The Event Log Object dialog box reappears with the new entry listed. You can edit the attributes of an existing

alarm by clicking on the **Setting...** button. Any entries can be deleted by clicking on the **Delete** button.

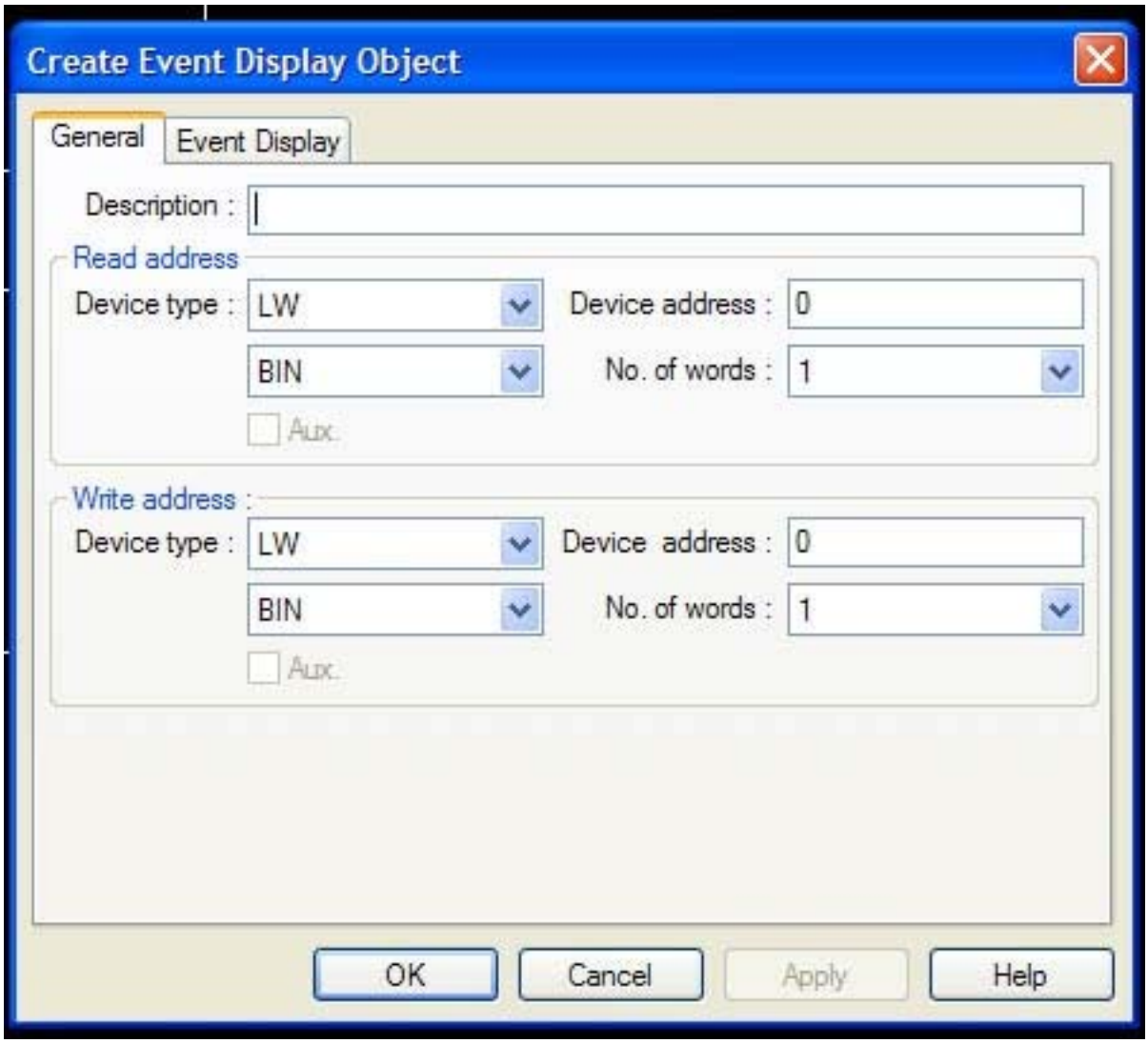
12. Click **Close** to close the Event Log Object dialog box. The Event Log Object is active no matter what windows are currently displayed on the HMI screen. Therefore, you do not have to place the object onto a window screen.

Displaying Events Using the Event Display Object

Though the Event Log Object continuously monitors the PLC for triggered events, it cannot display the events without the Event Display Object. The Event Display Object takes the event strings sent to it from the Event Log Object and displays them on a window screen as a list. The list contains all of the events that have occurred with the most recently triggered event at the top of the list. You do not have to display the entire list of events on a window screen. You can limit the viewable events to a 'scrollable window' that displays a given number of lines. You can then create a register that allows you to scroll through the list of events using the scrollable window. An acknowledge feature allows the HMI operator to touch an event recorded in the event list to display a popup window. The popup window can than be used to display further instructions to the HMI operator or to perform some operation.

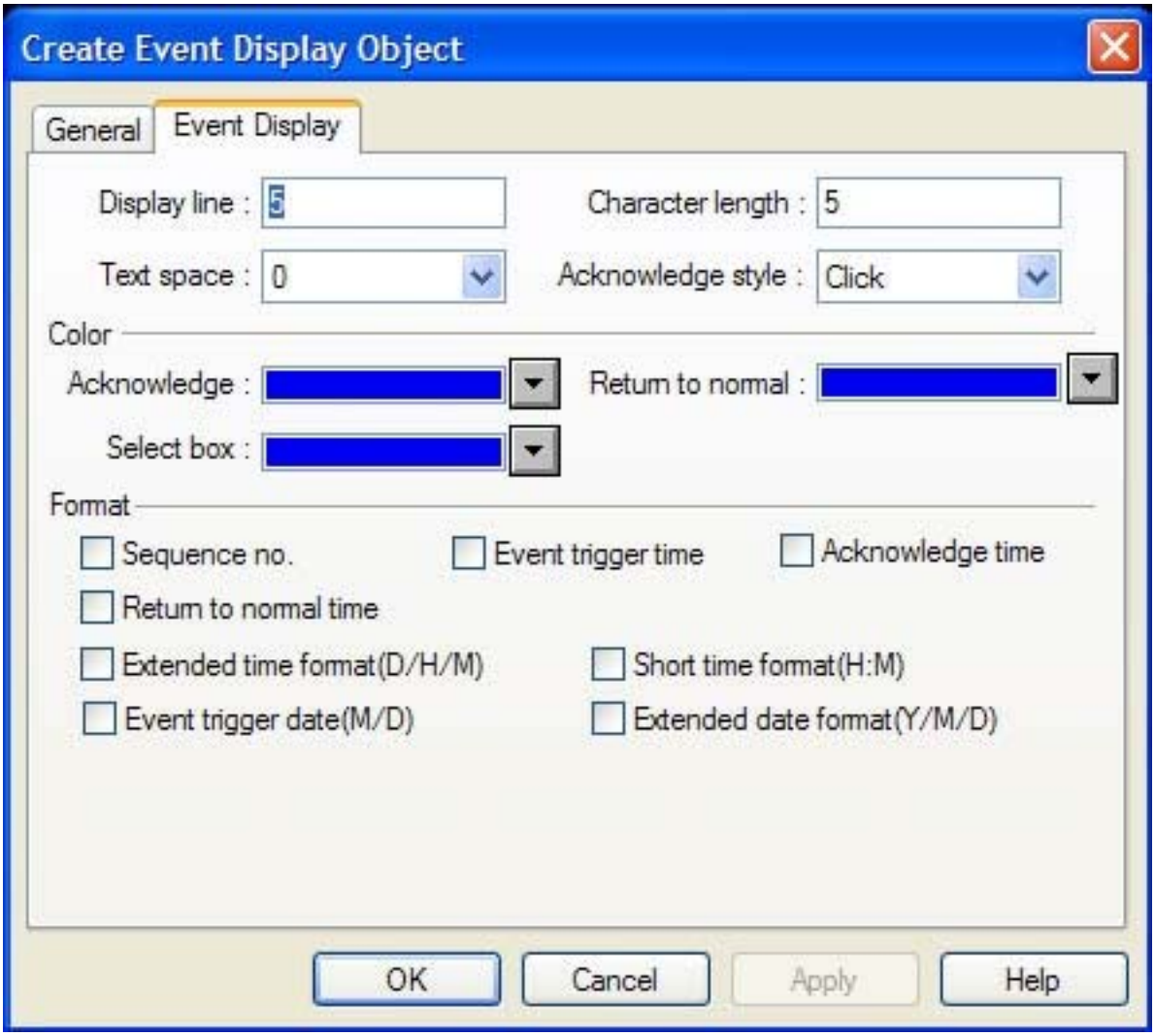
To create an Event Display Object

1. From the **Parts** menu, click **Event Display**. Or click the **Event Display** icon in the Part2 toolbar. The Create Event Display Object dialog box appears.



2. Use the **Description:** box to enter a title for the Event Display part. A description is not necessary but does help identify the purpose of the part.
3. In the **Read address** frame, select the 16-bit PLC register or HMI internal memory address that will be monitored by the Event Display part for the scrollable window. Select **BIN** (binary) or **BCD** format.
4. The **No of words** box is set at 1 and cannot be changed.
5. In the **Write Address** frame, select the 16-bit PLC register or HMI internal memory address that will be monitored by the Event Display part for displaying a popup window. Select **BIN** (binary) or **BCD** format. This address is written to by the Event Log Object using the Write Value number when the HMI operator acknowledges the event. **Note:** You must also configure an Indirect Window Object on the same window as the Event Display Object with the same write address in order to display the popup window.
6. The **No of words:** box is set at 1 and cannot be changed.

7. Click the **Event Display** tab to display the Event Display form.



8. Select the type of options the Event Display uses:

Function	Range	Description
Display line:	1 to 60	Selects how many events to display in the scrollable window. The number of lines refers to lines that are size 16 font. If the Event text string is size 24 font, then two lines are required for every event displayed.
Character length	1 to 80	Selects how long the scrollable window is. Select a length, which is equal to the maximum length of the largest event text string. If Size 24 font is used for the event text string, then two characters are required for each character in the text string.
Text space:	0 to 5	Select this option to insert extra rows of pixels between each event text string displayed. In other words, this sets the spacing between the lines of the scrollable window.
Acknowledge style:	Click or Double click	This setting determines what the HMI operator must do to acknowledge a triggered event. To acknowledge an event, the HMI operator must either click or double-click (press the same spot twice quickly) on the touch screen where the event string of the triggered event is located.

9. In the **Color** frame, select what type of colors the Event Display uses for some of the functions:

Function	Description
Acknowledge:	If the condition that triggered the event is still true, then this color is used for the event string when the HMI operator acknowledges or touches the string.
Select Box:	This color is used to highlight the event string that was last touched by the HMI operator.
Return to normal:	This color is used to indicate that the condition that triggered the event is no longer true.

10. In the **Format** frame, select the following attributes:

Attribute	Description
Sequence No.	If enabled, a sequence number will be displayed along with the event string whenever an event is triggered.
Acknowledge time	If enabled, this displays the time at which an event was acknowledged.
Event trig. time	If enabled, this displays the time at which an event was triggered.
Return to normal time	If enabled, this displays the time at which the condition that triggered an event is no longer true or active.
Extended time format (D/H/M)	When enabled, all times are shown as d:h:m. Day, Hour, Month.
Short time format (H/M)	When enabled, all times are shown as h:m. Hour, Month.
Event Trig. Date (M/D)	When enabled, the date that the event was triggered is added to each event as it is displayed. Month, Day.
Extended date format (Y/M/D)	When enabled, all times are shown as y:m:d. Year, Month, Day.

11. Click **OK** to close the Create Event Display Object dialog box. The main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.
12. Once the part is placed onto the window, you can adjust the attributes of the Event Display by double-clicking on the part.

Using Events for an Alarm History

Events are a complicated feature to understand and it may be a little unclear as to how events can be used. So let's through an application example in which we will use the events feature to create and display an alarm history. The alarm history shows all alarms that may have occurred in the past. The Alarm objects described earlier in this chapter cannot be used to create a history since the alarm strings are removed from display once the alarm condition no longer exists. But we may want to know what alarms have occurred in the past even though they no longer exist. Events can be used for just such a purpose.

For this example, we will use the same alarms that were shown in the section earlier in this chapter that described using the Alarm Display Object. Therefore, we will create five events which trigger from internal memory coils LB0-4.

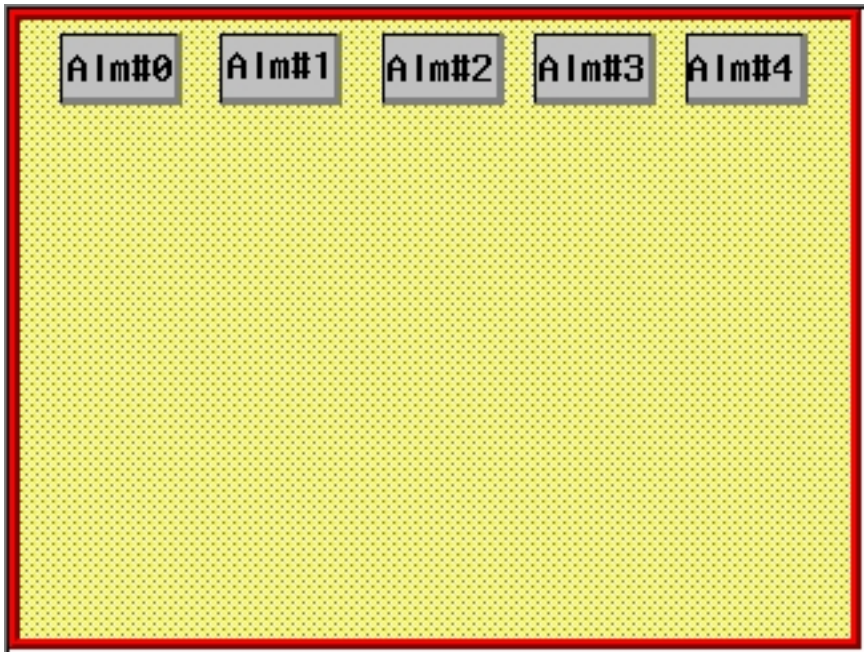
Create five Set Bit Objects to trigger the events

1. Open a new project with the startup screen displayed.

2. Create five Set Bit Objects with the following parameters. Refer to Chapter 8 *Representing Data with Graphics Objects* if necessary, for more information on how to use the Set Bit part.

Tab	Attribute	Setting
General	Write Address	LB0 to LB4
	Style	Momentary
Shape	Library	button1
	Item	0
Label	Attribute Color	Select color
	Align	Left
	Font	16
	State	0
	Content	Alm#0 to Alm#4
	Use Label	checked
	Tracking	checked
Profile	Width	46
	Height	27

3. Place the five objects on the top of the window:



4. These buttons will be used to simulate alarm conditions.

Create a text box and two Set Word objects to scroll event window

1. Create a Text Object that says “Alarm History”.

2. Create a Set Word Object with the following attributes:

Tab	Attribute	Setting
General	Write Address	LW0, BIN
	Attribute Style	Sub value (JOG-)
	Attribute Dec Value	1
	Attribute Bottom Limit	0
Shape	Library	button4
	Item	28
Label	Use Label	not checked
Profile	Width	41
	Height	20

3. Create another Set Word Object with the following attributes:

Tab	Attribute	Setting
General	Write Address	LW0, BIN
	Attribute Style	Add value (JOG+)
	Attribute Inc Value	1
	Attribute Upper Limit	10
Shape	Library	button4
	Item	32
Label	Use Label	not checked
Profile	Width	41
	Height	20

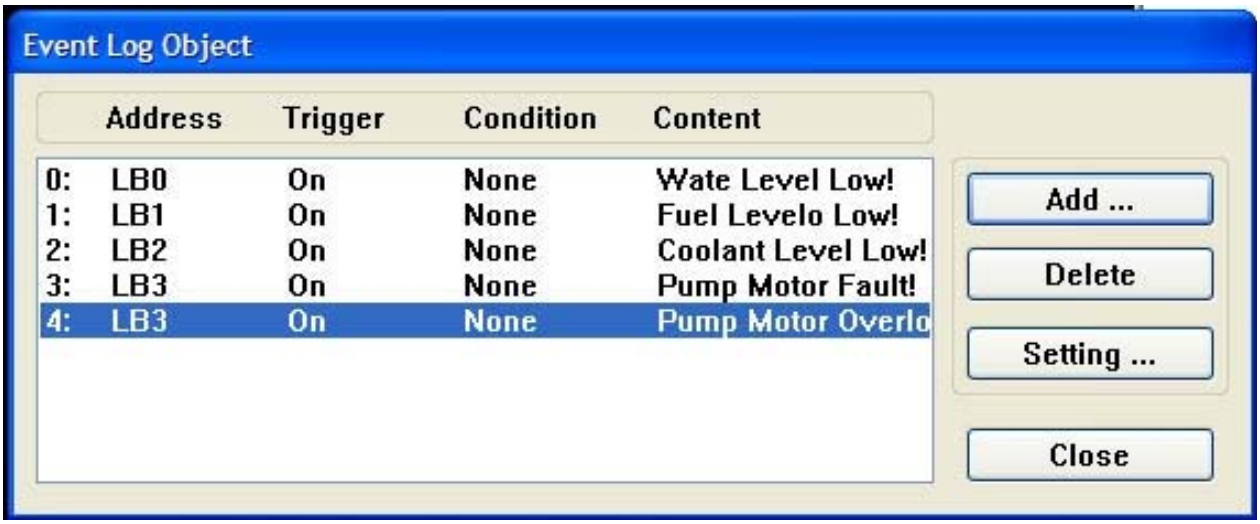
Place these objects onto the screen as shown:

Create an Event Log object to monitor alarms

1. From the **Parts** menu, click **Event Log**. Or click the **Event Log** icon in the Draw toolbar. The Event Log Object dialog box appears.
2. Enter the following parameters for each event:

Attribute	Setting
Address type:	Bit
Read Address	LB0 to LB4
Event trig	On
Text	Water Level Low! (LB0)
	Fuel Level Low! (LB1)
	Coolant Level Low! (LB2)
	Pump Motor Fault! (LB3)
	Pump Motor Overload! (LB4)
Font:	16
Color:	Select any color
Write value:	0

3. Click **OK**. The entries should look like this:



Create an Event Display object to display alarm history

1. From the **Parts** menu, click **Event Display**. Or click the **Event Display** icon in the Part2 toolbar. The Create Event Display Object dialog box appears.

2. Enter the following parameters for each event:

Tab	Attribute	Setting
General	Read Address	LW0, BIN
	Write Address	LW1, BIN
Event Display	Display Line:	3
	Character length:	31
	Text Space:	0
	Acknowledge Style	Click
	Color: Acknowledge	Any color
	Color: Return to Normal	Any color
	Color: Select box	Any color
	Format	Sequence No.

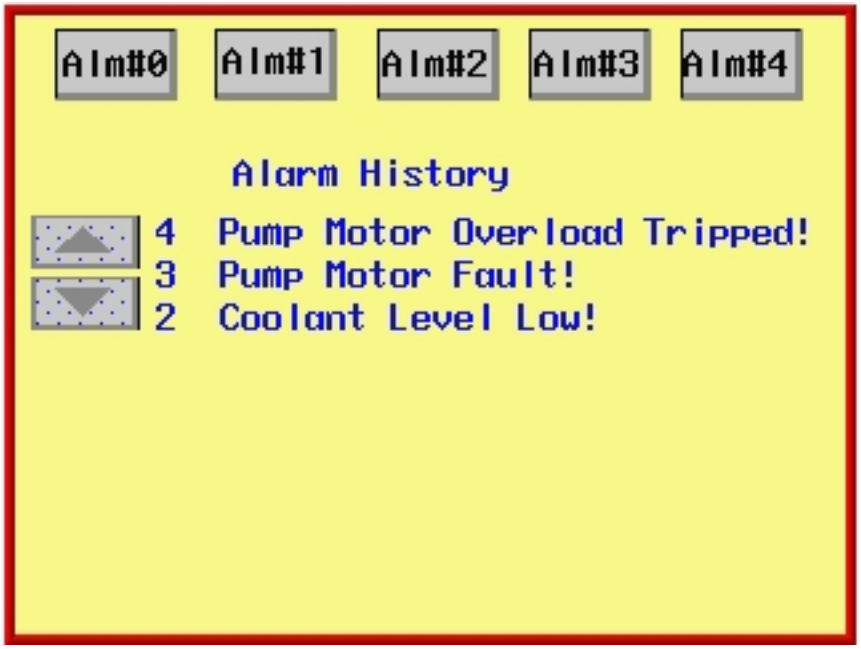
3. Click **OK**. Place object directly under the “Alarm History” text box.

Using the Alarm History

Save and compile the project, then download the project to the HMI or run the simulation mode. The startup screen should display:

Press each alarm button. Note that as each alarm is activated, the alarm history window displays the most recent alarm on the top of the list.

Since we have configured the window to be three lines, only the last three alarms are seen.



Use the scroll buttons on the left to see the two oldest alarms at the bottom of the list. Each item on the list will remain on the list permanently. The alarm message will remain on the list, even after the alarm condition goes back to normal. The sequence number is handy since it shows the total number of alarms that have occurred in the past. If you press the alarm buttons again, more messages will be added to the list.

Chapter 12 - Data and Recipe Transfer Objects

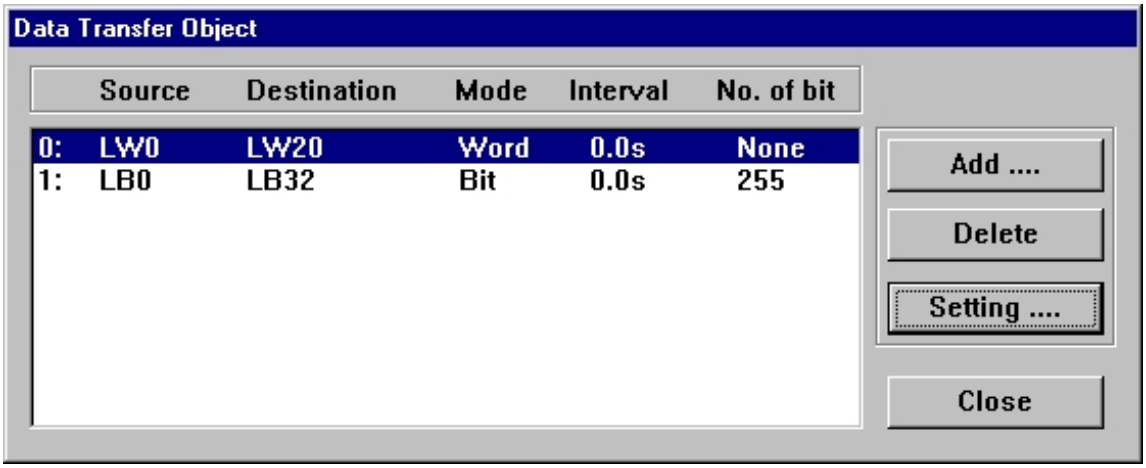
Easybuilder includes a Data Transfer object and a Recipe Transfer object, which enable data to be copied from one or more registers located in the PLC or the HMI to other registers. The Data Transfer object is used to continuously transfer data at a predefined rate. The Recipe Transfer object is a touch screen object that transfers data when activated by the HMI operator.

Using the Data Transfer Object

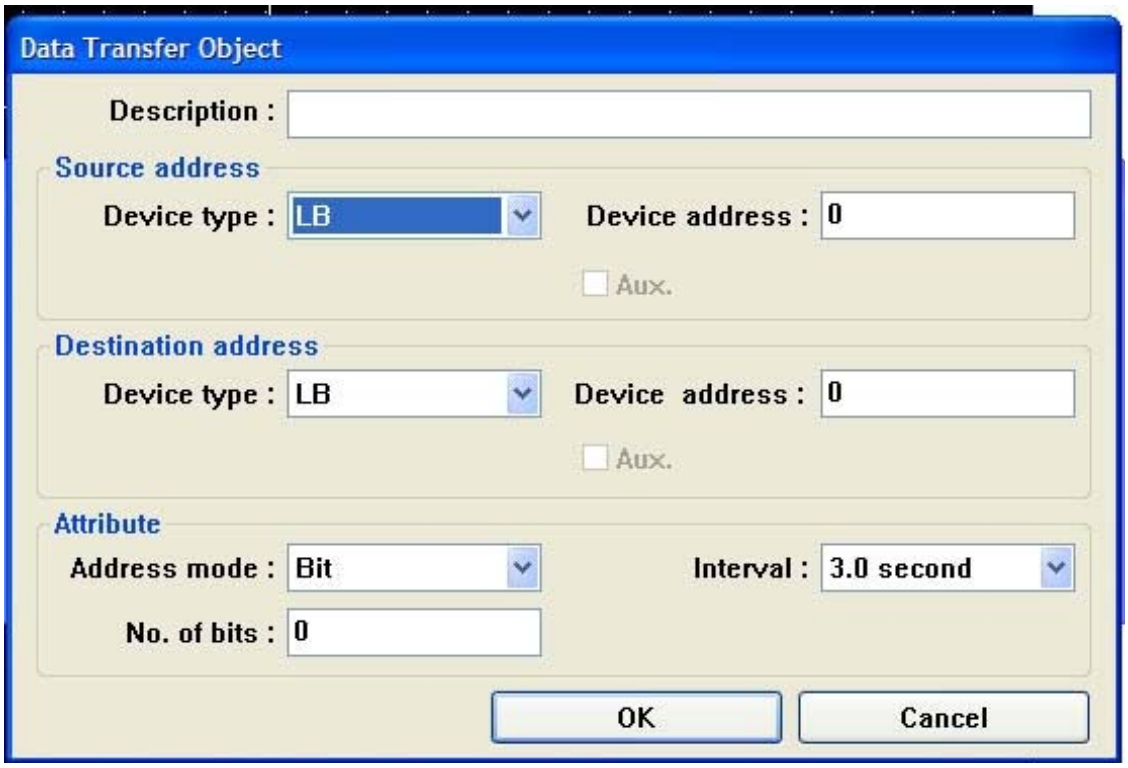
In some projects you create, you may need to transfer large blocks of data from PLC/HMI registers to other PLC/HMI registers. The Data Transfer Object is used to copy data from one or more PLC registers or HMI internal memory to other registers continuously on a timed interval. There is no limit to the number of Data Transfer Objects that you can create other than the HMI's available memory. Each Object created can transfer from 1-16 continuous registers or 1-255 coils.

To create a Data Transfer Object 

1. From the **Parts** menu, click **Data Transfer**. Or click the **Data Transfer** icon in the Draw toolbar. The Data Transfer Object dialog box appears.



2. Click the **Add...** button. The Data Transfer Object dialog box appears.



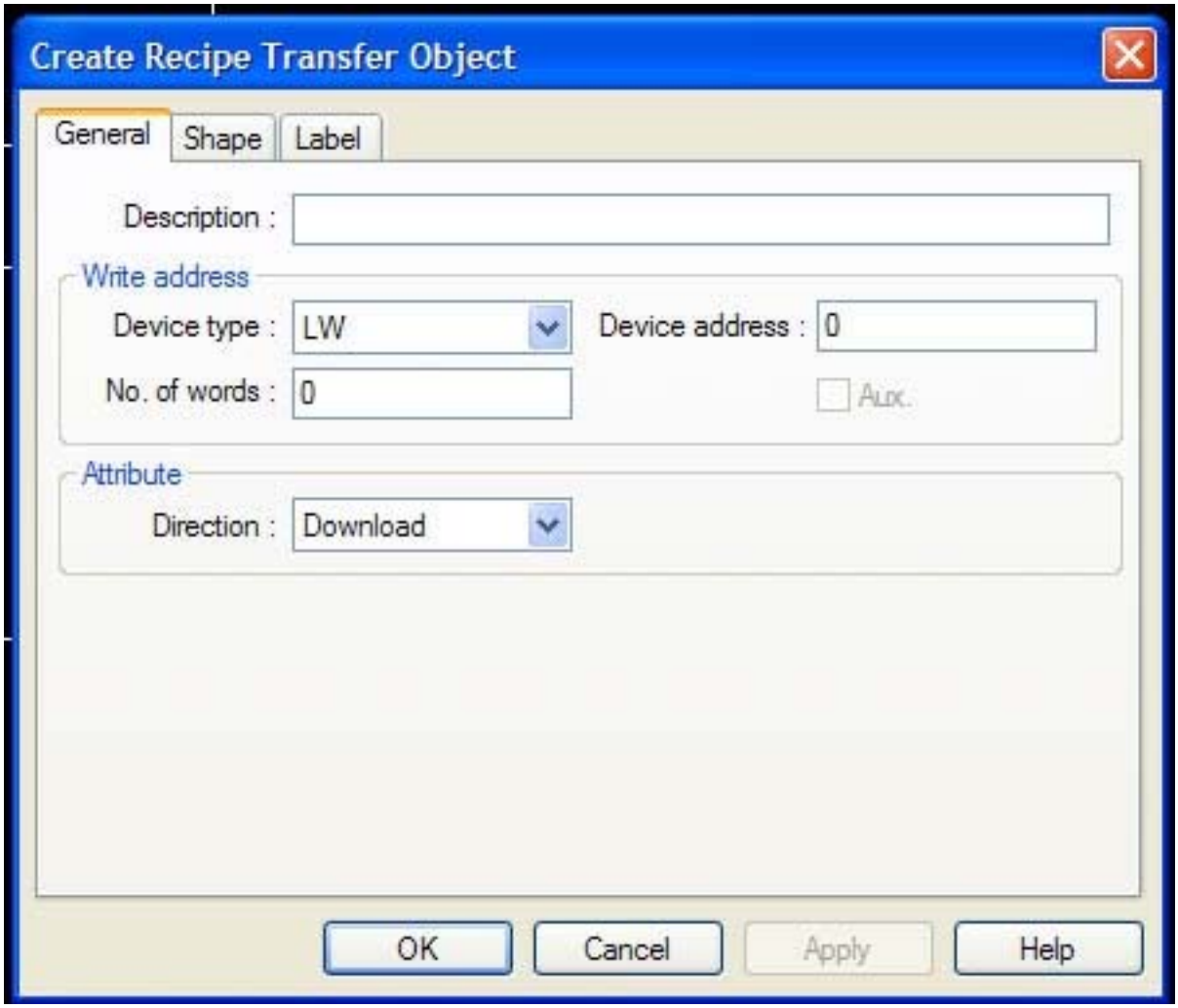
3. Use the **Description:** box to enter a title for the Data Transfer part. A description is not necessary but does help you identify the purpose of the part.
4. In the **Source address** frame, select the PLC coils or registers that are to be copied. If the **Address mode** is Word, then the **No. of words** box is used to specify the total number of 16-bit registers to copy. The range is 1 to 16.
5. In the **Destination address** frame, select the PLC coils or registers that receive the data.
6. In the **Attribute** frame, for the **Address mode:** setting click **Word** or **Bit**. Use the Word setting for data transfer of 16-bit registers. Use the Bit for data transfer of coils. If Bit is selected, enter the total number of bits that are to be transferred in the **Number of bits:** box. The range is 1 to 255.
7. The **Interval:** setting in the **Attribute** frame, is used to select the period at which the data is transferred. Select 0 for continuous data transfer. Range is 0 to 25.5 seconds in 0.1 second increments. Note that selecting a smaller period will decrease the amount of HMI processing time that can be allocated to other tasks.
8. Click **OK** to close the Data Transfer Object's Attributes dialog box. The Data Transfer Object dialog box reappears with the new entry listed. You can edit the attributes of an existing Data Transfer object by clicking on the **Setting...** button. Any entries can be deleted by clicking on the **Delete** button.
9. Click **Close** to close the Data Transfer Object dialog box. The Data Transfer Object is active no matter what windows are currently displayed on the HMI screen. Therefore, you do not have to place the object onto a window screen.

Using the Recipe Transfer Object

The Recipe Transfer Object is used to download or save data from one or more PLC registers to the HMI's internal registers that are reserved for recipes. This occurs when the object is activated by a press from the HMI operator. Each Object created can transfer from 1-32767 continuous registers!

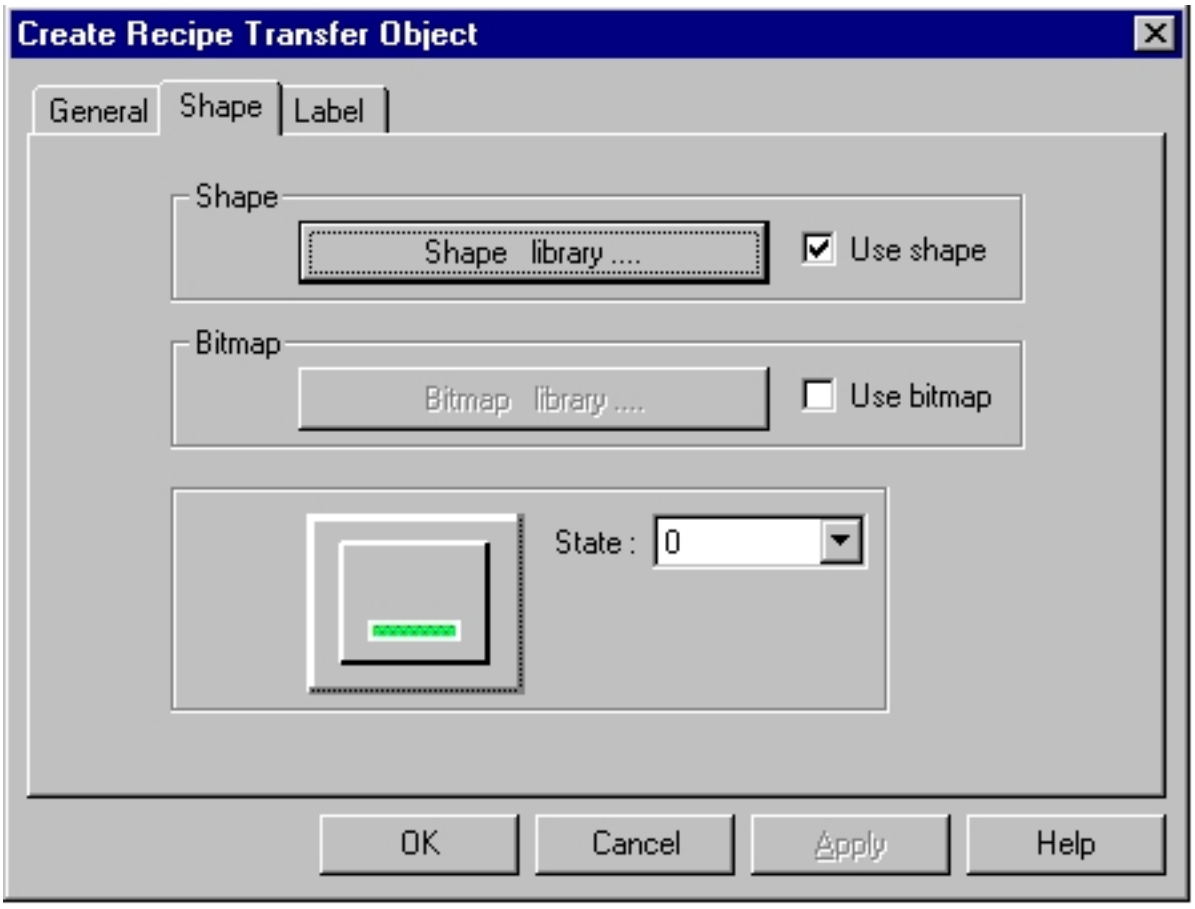
To create a Recipe Transfer Object 

1. From the **Parts** menu, click **Recipe Transfe**r. Or click the **Recipe Transfer** icon in the Part2 toolbar. The Recipe Transfer Object dialog box appears.

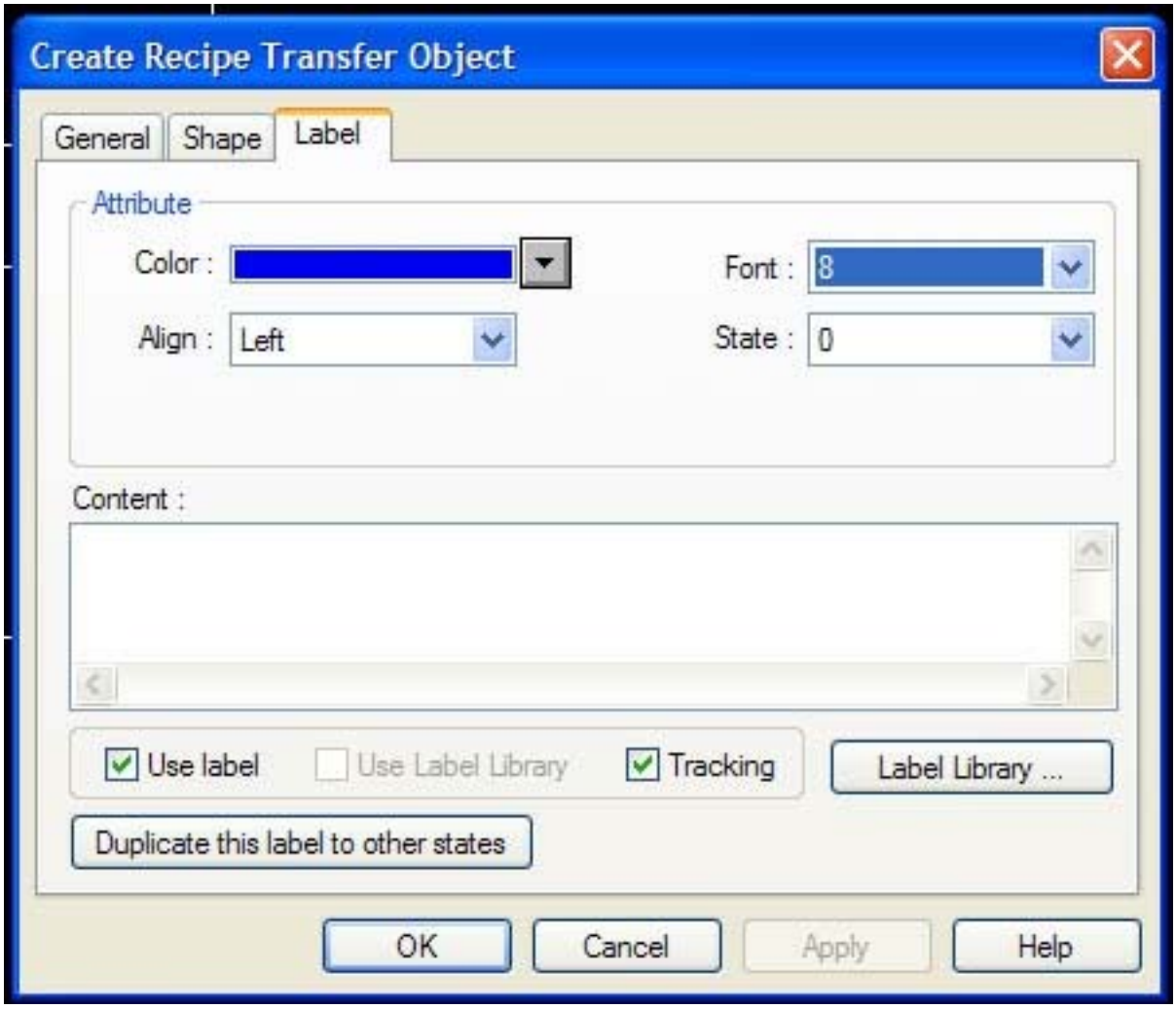


2. Use the **Description:** box to enter a title for the Recipe Transfer part. A description is not necessary but does help you identify the purpose of the part.
3. In the **Write address** frame, select the PLC registers that are to be saved or downloaded to. The **No. of words** box is used to specify the total number of 16-bit registers used. The range is 1 to 32767. The **Device Address** you specify for the PLC register is also the starting register used for the HMI's RWI (Indexed Address Recipe Word) data register.
4. In the **Attribute** frame, for the **Direction:** setting click **Download** or **Save**. Use the Download setting to send data stored in the HMI's RWI data registers to the specified PLC registers. Use the Save setting to retrieve data stored in PLC registers to be stored in the HMI's RWI data registers.

5. Click the **Shape** tab to display the Shape form.



7. Click the **Label** tab to display the Label form.



8. Click the **Use Label** checkbox if you want to use a label.
9. You can have a different label for each state of the Bit Lamp. In the **State:** box, enter the state that you wish to edit, then enter the label for that state in the **Content:** box. Do the same for other states. Click the **Duplicate this label to other states** button if you want to use the same label for all states.
10. Enter the color that you want for the label in the **Color:** box. You can select a different color for each label state.
11. Enter how you want the label to be positioned in the **Align:** box. You can select a different position for each label state.
12. Enter the size of the label in the **Font:** box. You can select a different font for each label state.
13. Click the **Tracking** checkbox if you want the labels for all states to follow or 'track' with the movement of one label when it is moved after the Recipe Transfer object is placed onto the window.
14. Click **OK**. The Create Recipe Transfer Object form closes and the main screen of EasyBuilder appears with the cursor tied to a rectangular outline of the part you just created at the upper left corner of the screen. Move the part to the desired location on the window.

    Once the part is placed onto the window, you can adjust the location of the label inside the part by clicking once on the label. This will highlight the entire object.

15. Click on the label again. Now only the label is highlighted, allowing you to move it without moving the part.

If you double-click (click twice rapidly) then you will not highlight the label but rather enter the Recipe.

Creating a Recipe

A recipe consists of a number of registers using the RW or RB devices. These devices are in a non-volatile area of the HMIs' internal memory area. A recipe is created by using the HMI's built-in Parts to store data in the RW/RB devices, and then transferring that data to the selected PLC registers.

Recipe Devices There are 4 types of Recipe devices: Recipe Word (RW) , Indexed address Recipe words (RWI), Recipe Bit (RB), and Indexed address Recipe Bit (RBI). The usage of RW and RB devices is the same. Therefore, in the following discussion, R*n* refers to both RW and RB devices, and R*n*I refers to both RWI and RBI devices.

R*n* Device Refers to a direct R*n* address. That is, the value placed in the Address field of an R*n* device refers directly to the R*n* address. For example, R*n*100 refers directly to the data at address R*n*100.

R*n*I Device An address with an R*n*I device is an *Indexed* address. The address referred to by an R*n*I device is actually the specified address plus an indexed offset. The offset is the value in address LW9000. For example, suppose address LW9000 contains the value 50, and the Address field of an R*n*I device is set to 100. This *Indexed* address actually refers to the data at address R*n*150.

RB devices use the same memory area as the RW devices. The RB device allows Bit-level access to recipe data. The bit address is specified by using the RW address, followed by the bit number 0-F (hex). For example, Bit RB0 is Bit 0 in RW0; Bit RW2047F is Bit 16 in RW2047.

Rn Devices Vs RnI Devices

R*n*I devices Indexed address devices are most useful when it is necessary to 'page' through a number of recipes. R*n*I devices allow a single window to manage all recipe items. Conversely, the use of R*n* devices requires a separate window for each recipe, with each object on each window having a separate address. This not only uses more of the HMI's resources, but could also lead to configuration errors as objects are copied, pasted, and edited.

R*n* devices Use when it is desired to show unique information from a recipe. For example, it might be necessary to show, one window, the names of all of the recipes.

For an example of how to create recipes, please consult the EZware-500 online Help file.

Chapter 13 - Macros

Using Macros

The “macro” language in EasyBuilder allows the programmer to perform simple transfers of data and/or simple to complex conditional logic and transfers.

Macros are high-level language programs that can run inside the HMI to perform various data transfers and can perform conditional logic. They are useful for doing tasks that are not supported in a standard object or feature of the HMI. The macro language combines syntax elements of C, C++ and a little of its own style.

Previous experience in programming with C or C++ would be helpful when developing macros. It is not the scope of this document to explain concepts of high-level language programming and the basics such as: variable data types, arrays, conditional logic, and variable passing. However, it is the scope of this document to discuss the commands and command structures that are required in building macros.

What's new with macros in Version 2.6.2?

- **NEW:** The macro ‘name’ (instead of just its number) is listed in the PLC Control Object (‘Execute Macro’) Attribute dialog box and it is also listed in the list of PLC control objects.
- **NEW:** When the whole project is compiled (Tools→Compile), the macro size is listed separately from the other project memory-usage sections.
- **Fixed**: The GetData() function where it stored the wrong address.
- **Fixed**: If “extended mode” is selected under EDIT->System Parameters->{Editor} tab->Addressing, then the macro does NOT require that extended addressing format be used. If no extended addressing is put before the address (ie. 2#123), then the address selected is defaulted to the ‘PLC Station No.’ address in the System Parameters.
- **Bug:** If a 16-bit external register is plugged into (i.e. GetData) a macro’s INT declared variable, it will work in the simulator, but it will not always work when downloaded to the HMI. The work around is to change the declaration to a SHORT instead of an INT.
- **Bug:** The GetData() function does not always work correctly when loading values into an array. If the starting element in the array is 0 or a multiple of 16, then GetData() works correctly. If the starting element is any other value, the transfer ends at the first 16-bit boundary.

  In the following example, the local bits 300-363 are all set.
  GetData(result[0] ,LB_Binplc ,300,64)
  After execution, the first four bits in result[] are set, and the next 12 are cleared. The remaining bits (LB16-63) are all set as expected. To properly load result[], the following code is required:
  GetData(result[0] ,LB_Binplc ,300,4)
  GetData(result[4] ,LB_Binplc ,304,60)
  SetData() appears to work correctly.
- **Bug**: ‘Case’ statements appear to not work correctly as expected. Avoid using CASE statements. Use nested ‘If…Then’ statements instead.
- **Bug**: A ‘bool’ declared variable does not function properly in conditional or logic statements. It is best to declare ‘bool’ type variables as ‘ints’ instead.

What's New in Version 2.6.0

- **NEW**: The register type now ends with “plc” or “aux” tacked on the end, for example: **GetData(variable, 4x_Bin, 123,1)** is now: **GetData(variable, 4x_Binplc, 123,1)**. “plc” is used if the register is used through the main serial ports or Ethernet port. “aux” is used if the register is obtained through the auxiliary port.

- **NEW**: If "extended mode" is selected under EDIT->System Parameters->{Editor} tab->Addressing, then the macros ***require*** that extended addressing format be used, otherwise you will get compiler errors. The format to use is **NodeAddress** + # **sign** + **address**, for example: **GetData(variable, 4x_Binplc, 1#123,1)** to address node #1, and address 123.
- **Fixed:** the ability to utilize LW:9000 as a pointer for using RWIs (Recipe Word Indexed) registers. RBIs still can not utilize the LW:9000.
- **Bug**: Inline conditional logic [ie. If **`(x==1 AND y==2 AND z==3 Then …)`**] will not work unless the variables contained in the conditional statements are defined as 'int's.
- **Bug:** If a GetData() uses an array of BOOL that does not start on address [0] or [16] or [32] or some other number that is divisible by 16, then the bits can get stored to the wrong locations.
- **Bug:** the 4th parameter of a GetData() or a SetData(), is the number of elements to read or write. This used to allow a number up to 255, it now will only allow a number up to 127.
- **Limitation:** Macros that use extended addressing will only address nodes 0 to 7. They used to address up to 255 nodes.

What's New in Version 2.5.2

- **New**: Comments can now be added after any command line by using a double-slash '//' before the comment, for example:
  For n = 100 Down 0 Step 2 // Cycle through variables
- **Bug:** The macros do not have the ability to utilize LW:9000 as a pointer for using RWIs (Recipe Word Indexed) and RBIs (Recipe Bit Indexed) registers.

Macro Sample and Implementation

Macros can be used for customizing an application or when a special operation is needed that the normal HMI objects (set bit, data transfer, etc) do not provide. A few possibilities are listed:

- Unusual or non-linear scaling of data
- Displaying the result of arithmetic operations on 2 or more PLC register values
- Indirect addressing (where a PLC register holds the address of the desired data)
- Lookup tables
- Transferring Data once, instead of periodically
- Writing data to the PLC in a specific sequence
- Timed splash screens

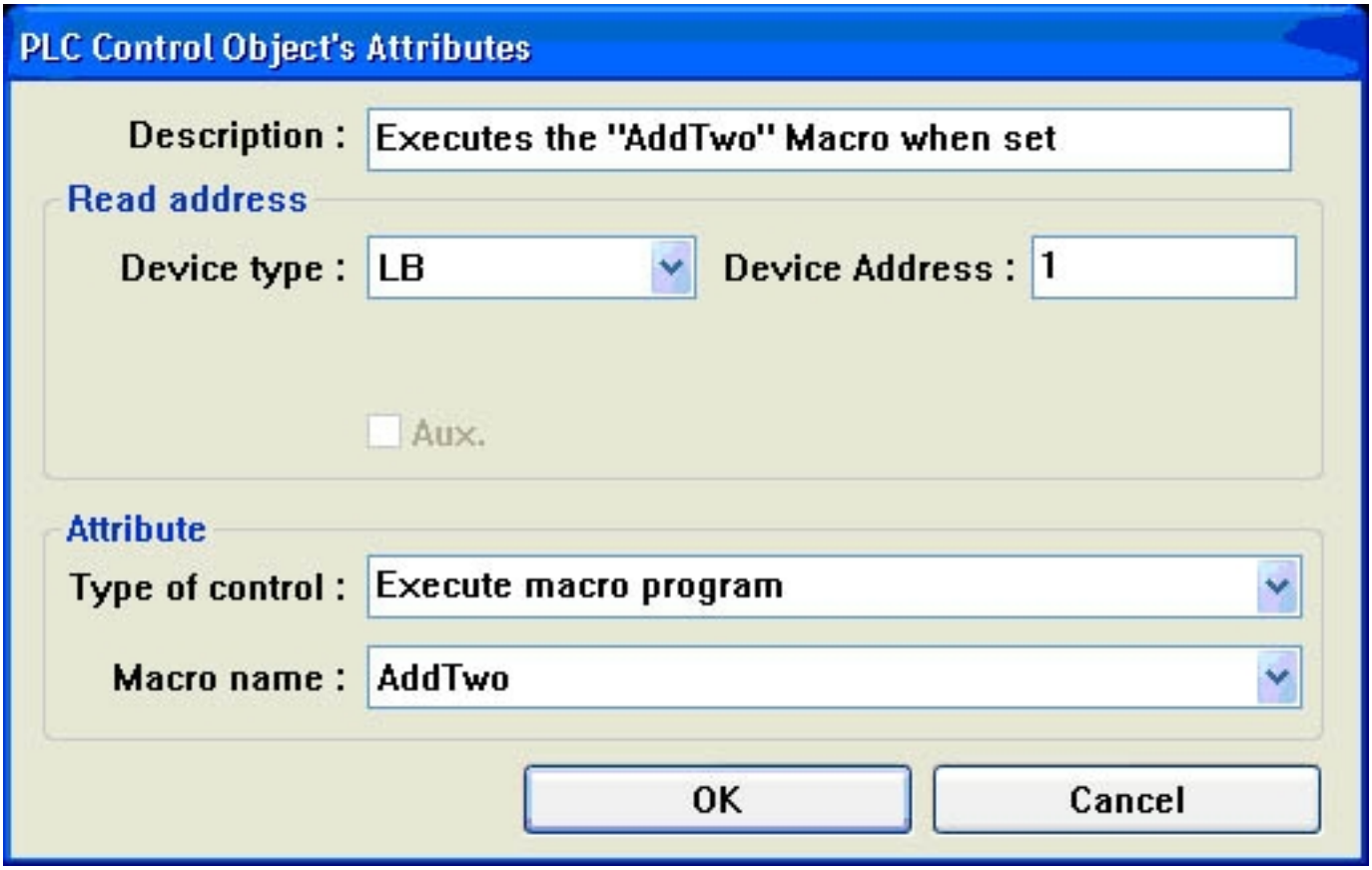
A macro is written using the "macro" utility found in the 'Tools' menu list of Easy Builder. The 'Add' button will create a new worksheet to program a new macro. The programmer first gives the macro a name in the top right corner. Then programming can be added. After the code is finished, it needs compiled. If the code compiles correctly there will be no errors and the macro name will be added to the top list. If the macro does not compile correctly it will display error codes. If the programmer should cancel out from creating the macro before a successful compile, it will put the macro into the bottom panel of the macro list and it will not be available as a usable macro, but the text will remain to be edited later.

The macros are triggered by a setting a bit. The bit is specified in the 'PLC Control Object' list. The bit may be a local bit 'LB' or it may be a PLC bit. Each macro created will have a unique name and bit.
When a specified bit is set high so it executes a specific macro, the macro will be entered into and its variables initialized. The macro will run its sequence of events and will exit when finished. The bit that called the macro into execution should be cleared by the macro before it exits, otherwise the macro will immediately be entered again upon exit. If this immediate re-entry should happen, this could form an endless loop and will certainly cause a taxing of the CPU time and cause slow screen refreshing and communications.

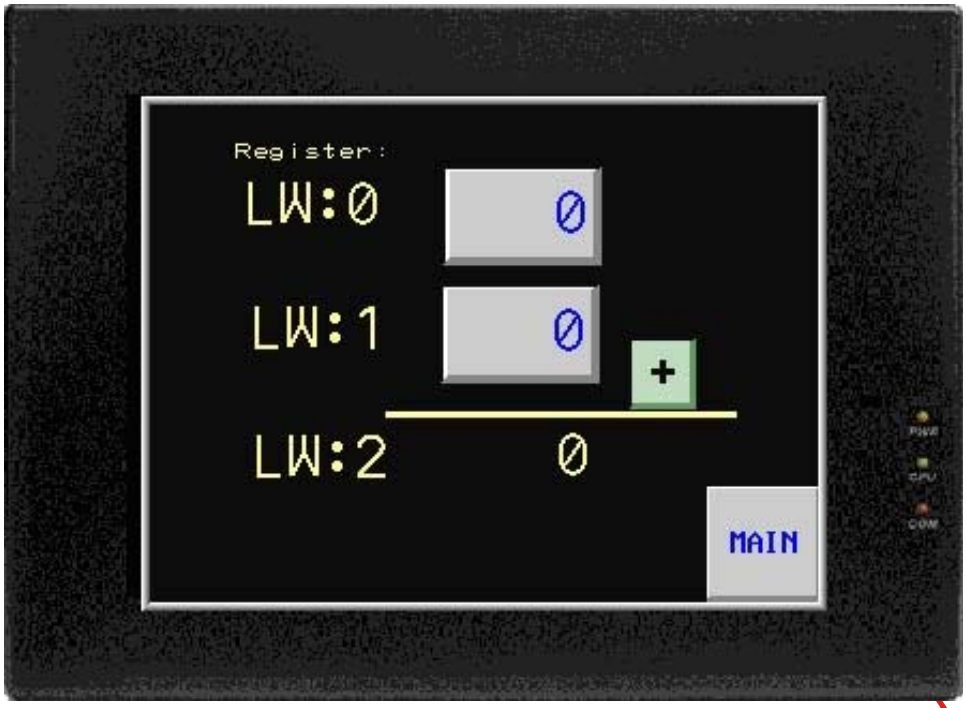
A sample macro is listed below which reads two local word registers and adds them together and stores them into another register location:

```
Macro_Command main(  )
      short num1
      short num2
      short result
      bool AllDone = 0
      GetData(num1 ,LW_Binplc ,0,1)    // get two registers to
      add (LW:0 & LW:1)
      GetData(num2 ,LW_Binplc ,1,1)
      result = num1 + num2             // add the two registers
      SetData(result ,LW_Binplc ,2,1)  // store the result (LW:2)
      SetData(AllDone ,LB Binplc ,1,1) // done with the macro,
      clear the bit     //  that called it. (LB:1)
      End Macro Command
```

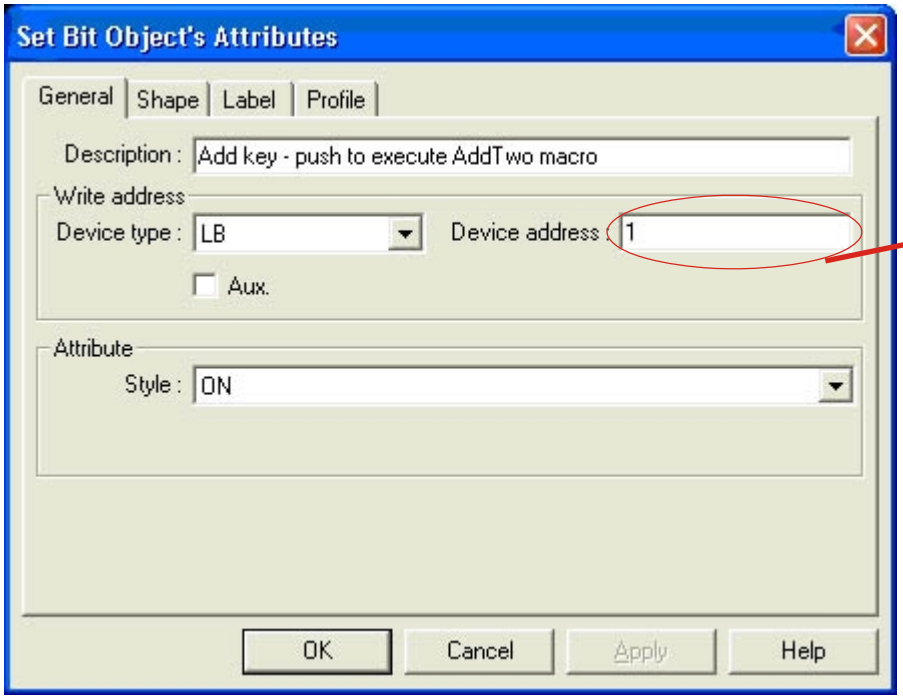
The macro was named “AddTwo” and the bit that executed this macro was LB:1. After the macro was written, a ‘PLC Control Object’ was added where the ‘Type of control’ was selected first to be ‘Execute macro program’. Then the device type showed what bit types were available. A ‘LB’ was selected and the address ‘1’ was entered. Finally, the macro name was selected (in older software, you would select the macro number instead of the name). Below shows the PLC control object attribute window.



PLC Object's Attributes



Project Example of Add 2



Set Bit Objects Attributes Dialog

In the example above, a set-bit object was placed on the screen as a '+' button. When this button is pushed, it forces LB:1 to ON and the 'AddTwo' macro executes, storing the result in LW:2 and turning LB:1 to OFF and the macro stops.

Variables, Declarations and Memory Usage

Variables, constants & functions are named by the programmer. Variables and constants are called *operands*. Variable names can be any character or number (a-z, A-Z, 0-9, or '_') and can be as many characters as you need to describe it (the compiler has been tested to variable name lengths of 32 characters, but it is best to keep the variable names small for ease of reading the code). Variable names are not case sensitive, so 'result' is the same as 'ReSult' or 'RESULT'. Variable names must always start with a letter (i.e. variables cannot begin with a number or the underscore '_' character).

Memory Usage:

Macros use a minimum of about 2.5K, plus the memory required for each variable type:

- float: 4 bytes (32-bits)
- int: 4 bytes (32-bits)
- short: 2 bytes (16-bits)
- char: 1 byte (8-bits)
- bool: 1 bit
- arrays: the corresponding number of bytes for each element, plus another 2 bytes for the index

Macros use memory from the background task (defined as *Window 0*) area. There is a default memory size of 320K available for macros (only 220K is allotted if the 'Fast Selection' task bar is enabled) and the there are other background task objects that also use memory in this area, such as:

- Trends
- XY Plots
- Data Transfers
- Alarms
- Events
- PLC Controls
- Printer functions

If the application exceeds the 320k limit, both the simulator and the HMI will display a System Severe Error message. To check the memory usage; right-click on the Simulator window and select System Resource from the menu. To add more memory to this area, reduce the number of total pop-ups possible. The setting is in the system parameters {General} tab. The default is 6; change this to 4 and the memory will increase (be re-allocated) for window 0 (background task) functions.

Variable Declarations

Each variable that will be used in the macro needs to be declared as a specific 'type' of register. The declarations are listed as the first part of the macro. Any declarations contained within the macro function is considered 'Local' and any variable outside the macro function is considered 'Global'. Local variables are only seen within the function they are declared in. Global variables retain their values and can be globally used by all functions. Below, are listed the various data types that can be declared:

TypeDescription

```
Float
Single-Precision Floating point variable (32-bit signed,
IEEE-754 format)
Int
Integer variable (32-bit signed)
Short
Short integer variable (16-bit signed)
Char
Character variable (8-bit unsigned)
Bool
Boolean variable (1-bit)
```

Variables inside macros are initialized to all '1's as a default (i.e. 0xFFFF), so don't assume they are zero values when you enter the macro. It is good programming practice to initialize variables during declaration, or use assignment statements before they are used.

Variable Initialization

Initialize a value of variable in the declaration statement directly. (e.g: int RPM = 75) Use the assignment operator (=) to initialize a value to the variable. Variables can be declared and initialized in the following manor: Stacked Example: Inline Example (separate like-types with a comma):

```
short a = 0
short b = 0
short c = 0
short a=0, b=0, c=0
```

Variables contain an unknown, random value when declared. Variables must be initialized before they are used.

Array initialization

Formats:

```
int MyArray[10] = {1,2,3,4,5,6,7,8,9,10}
char LetterArray[6] = 'MYWORD'
```

The initial values are written within the brackets {} and are divided by comma (,). These values are assigned in order from left to right starting from array index=0.

Constants:

A constant is a numeric or boolean value that does not change. Constants may be written as any of the following:

	Written As	Examples
Decimal constant	1234	short MyVal = 1234
Hexadecimal constant	0xFA20	short MyVal = 0xFA20
ASCII code (character constant)	'ABCD'	char String[4] = 'ABCD'
Boolean: True (not zero), False (zero)	True, 1, False, 0	bool Done = 0, or, bool Done = False

Reserved Words

The following symbols and names are keywords that are reserved for use by macros. They cannot be used (as a complete name) in any function name, array name, or variable name. However, the reserved words may be contained within a variable name such as: my_int, TheEnd, etc.

And	down	int	Step
bool	Else	next	Then
Break	End	not	To
Binaux	False	Or	True
Binplc	float	return	void
Case	For	select	wend
char	GetData	SetData	While
Continue	If	short	xor

Operator

Below is a list of the recognized operators.

Group	Name	Sym
	Assignment:	=
Assignment operator:	Addition:	+
	Subtraction:	-
	Multiplication:	*
	Division:	/
	Modulo:	%
Comparison operators:	Less than:	<
	Less than or equal to:	<=
	Greater than:	>
	Greater than or equal to:	>=
	Is equal to:	==
	Not equal to:	<>
Logic operators:	Conditional 'AND':	AND
	Conditional 'OR':	OR
	Exclusive 'OR':	XOR
	Boolean 'NOT':	NOT
Bitwise and shift operators:	Left shift:	<<
	Right shift:	>>
	Bitwise 'AND'	&
	Bitwise 'OR'	|
	Bitwise 'XOR'	^
	Bitwise complement:	~

Order of Precedence

The process order of many operators within an expression is called the ‘order of precedence’. Priority of the same kind of operator (From left to right, from top to bottom)

Arithmetic operator: ^ (* , /) (%) (+ , -)

Bitwise & Shift operators: From left to right within the expression

Comparison operator: From left to right within the expression

Logic operator: Not And Or Xor,

- Arithmetic operator is higher priority than a Bitwise operator
- Bitwise operator is higher priority than a Comparison operator
- Logic operator is higher priority than an Assignment operator

Main Functions and Sub-functions

The macro must have one and only one "**Macro_Command main()"** function which is the execution start point of any macro file. Any sub-functions must be pre-defined and written before the **main()** function.

The format is:

```
Macro_Command main( )
// The main macro code goes here
End Macro_Command
```

Any other sub functions must be declared before the main() function can use it, for example:

```
Sub SQR(int MyVar )
   // sub routine code goes here
End Sub
Macro_Command main( )
 // The main macro code goes here
   Result = SQR(MyNum)  //The sub function "SQR()" is
called
End macro command
```

Local and Global Variables

Local variables exist only within the function in which they are defined.
Global variables exist throughout the entire macro, and are available for any function in the macro. NOTE: If a local variable has the same name as a global variable, the compiler will use the local variable in the function instead.

Global variables are global only within the macro file in which they exist. It is not possible to share variables between different macro files. To share data between macro files, read/write the data to share into the HMI's Local Word (LW) and/or Local Bit (LB) addresses using the macro's GetData() and SetData()UM_PLC functions.

Creating Variable Arrays

Arrays are fixed-depth and 1-dimensional only. Arrays can contain up to 65,535 elements. The ***type*** is the declaration of the array of elements, such as an array of *ints*, or *shorts*, or *bools*, etc. The ***array_size*** is the number of elements to contain in the array, such as an array of [10] would contain 10 elements of the declared ***type***. The first register in an array always begins with the 0^{th} element and the last register is the declared number of elements minus 1.

The format is: ***[type] Array_Name[Array_Size]***

For example:

```
char table[100]
```

The above declaration defines an array of 100 type 'char' registers named 'table'. The first element in table is address 0 and the last is 99, so it would be used as:

```
char FirstElement
char Last Element

FirstElement=table[0]
LastElement=table[99]
```

Using Macros Within Recipes

The macro utility can easily use the recipe memory area. There are 60,000 recipe word registers that are available to the user for general use (addresses RW:0 to RW:59999). Recipe memory at address RW:60000 and above are reserved and have special functions. See the Easy Builder Help section or view technical note #TN1099 for the current reserved memory listing.

Register types: RW, RWI, and the pointer LW:9000 can be used to read or write the recipe memory area. 'RW' registers are used to directly address a recipe register. 'RWI' registers are used to indirectly address a recipe register by using the pointer in local word memory LW:9000. If indirect addressing is used, then the direct recipe address is determined by adding the value contained in LW:9000 to the address of the RWI register. For instance, to indirectly point to the direct recipe address RW:20, an example would be that the macro would store the value 15 into the LW:9000 pointer and then use an RWI address 5 (RWI:5) to GetData() or SetData() {LW:9000 = 15 + RWI:5 → RW:20}.

The pointer has a maximum limitation of 32767, so LW:9000 can indirectly address from an offset of 0 to 32767. To indirectly address recipe registers above 32767, you will need to have the RWI address set to a higher address.

The macro code on the next page will clear all general-use recipe memory area. Because of the pointer's limitation (max of LW:9000 = 32767), it breaks up the incrementing of memory into two sections, the lower half (0 to 29999) and upper half (30000 to 59999).

```
// Macro clear memory routine
Macro Command main(  )
int   Address = 0
short Initial = 0
short Done    = 0
short Section = 0
// LW:2 is used as an indicator in the project
SetData(Section, LW_Binplc,2,1)            // LW:2 = 0 (Lower
Half)
For Address = 0 To 29999
SetData(Address, LW Binplc,9000,1)        // LW:9000 incr.
through memory
SetData(Initial, RWI Binplc,0,1)          // RWI:0 indirect
address is Low

Next Address
Section = 1
SetData(Section, LW Binplc,2,1)            // LW:2 = 1 (Upper
Half)
For Address = 0 To 29999
SetData(Address, LW_Binplc,9000,1)        // LW:9000 incr.
through memory
SetData(Initial, RWI_Binplc,30000,1)      // RWI:0 indirect
address is Upper
Next Address
SetData(Done, LB_Binplc,1,1)               // Turn off the bit
that called the macro
End Macro Command
```

Using 32-bit Registers Within Macros

When reading or writing a 32-bit (2-word) register that is external to the macro using GetData() or SetData(), you must use an array of (2) 16-bit 'shorts'. Internally, the macro can use an 'int' type variable to store a value up to +/- 2,147,483,647 (a signed 32-bit register). But when using the GetData() or SetData() functions it only does a 16-bit external transfer even though it may be declared as 'int' (32-bit).

```
// This routine writes Hex value 0x12345678 to a 32-bit holding
register
short Data[2]
int value = 0x12345678
// split 32-bit value into (2) 16-bit values
 Data[1] = (value & 0xFFFF0000) > 16    //store the MSW
Data[0] = value & 0x0000FFFF            //store the LSW
// store the (2) 16-bit values into LW14 and LW15
SetData(Data[0] ,LW Binplc ,14,2)
```

```
// This routine reads a 32-bit holding register
short Data[2]
// grab a 32-bit register [(2) 16-bit] value from a PLC register
address 4x:1
GetData(Data[0] ,4x Binplc ,1,2)
```

```
// This routine puts (2) 16-bit array variables into one 32-bit
variable
Int LVal
short Data[2]
LVal = (Data[1] < 16) + Data[0]
```

Using Floating Point Registers Within Macros

Reading and writing external floating point registers can only be used in a macro as a transfer function. The macro routine has no functionality to interpret an external floating point formatted (IEEE-754 format) register. Although a variable can be declared as a 'float' type within the macro, it can not store the external float register properly.

When reading or writing a 32-bit floating point (also call a 'real') register that is external to the macro using GetData() or SetData(), you must use an array of (2) 16-bit 'shorts'. Internally, the macro can use an 'int' type variable to store a value up to +/- 2,147,483,647 (a signed 32-bit register). But when using the GetData() or SetData() functions it only does a 16-bit external transfer even though it may be declared as 'int' (32-bit).

```
// This routine will not work – a GetData of more than one
element requires
// a variable that is declared as an array
float RealData
// get the floating point register and store it in 'RealData'
GetData(RealData ,F8 Binplc ,1,2)
```

```
// This routine reads a 32-bit holding register
short Data[2]// grab a 32-bit register [(2) 16-bit] value from a
PLC register address 4x:1 GetData(Data[0] ,4x Binplc ,1,2)
```

Statements, Conditions & Expressions

An ***expression*** combines constants, variables, arrays, functions, and operators under certain rules. Expressions can be logical or mathematical equations.
Below are listed some of the different styles of statements:

- A ***declaration*** (definition) statement is constructed like the following:

Format	Examples	Explanation
***type* name**	`short` `MyVar`	This defines the name of the variable register and its *type*
***type* name** **[constant]**	`Short` `MyVars[10]`	This defines an array name and the [constant] is the number of '*type* elements' in the array

- An ***assignment*** statement will transfer (assign) an expression to a variable. The form is:

Format	Examples	Explanation
variable = expression	`MyVar = 10` `MyVar = x + y`	This assigns the result of the expression to a variable

- A ***logic*** or ***conditional*** statement processes a logic expression and branches depending on the result. This assigns the result of the expression to a variable. The ending statement **'End If'** is written as two words, not one word. Note that there is a difference between the conditional equality noted by two equal signs '= =' (read as: 'is equal to') and the assignment equality noted by only one equal sign '='.

Format	Examples

If *Condition* Then
[*Statements*]
Else
[*else statements*]
End If
	`If x == 1 Then`
	`  y = y * 5`
	`  z = z – 12`
	`Else`
	`  y = y * 2`
	`  z=y-1`
	`End If`

	`If (x == 0 OR x == 1) AND y == 0 Then`
	`  z = 10`
	`Else`
	`  z=5`
	`End If`

- A ***control*** statement will branch the flow of the program expression. The [*TestExpression*] is required. A 'Case' statement will evaluate the '***TestExpression***' and determine a numeric value. The program will then test each '**case #**' that matches and will perform all the statements within its block. If a '***Break***' is reached, the program flow will jump directly to the 'End Select' statement, bypassing all other Case conditions. not one word.

Avoid using "Case" statements at this time, case logic is testing to be erratic

- If no case #s match, the program branch will execute the 'Case Else' statement(s).
- The ending statement 'End Select' is written as two words

Format	Examples
Select Case [TestExpression] Case - [statements-n] ... [Case Else [else statements] } End Select	Select Case Status Case 0 Return 1 Case 1 X=2 Break Case 5 Case 7 Return 8 Case Else Return 0 End Select

A l***ooping control*** statement is used for looping a pre-determined number of times. For counting up, use the keyword **To**. For counting down, use the keyword **Down**. Counter Necessary. The counter of looping control. It can be integer o character.

Start

Necessary. The initial value of Counter.

End

Necessary. The end value of Counter.

Step

Options. The increment decrement step of Counter. It can be integer an can be omitted when value is 1.

Statements

- Optional. Statement block between For and Next which will be executed.

Format	Examples
For counter = start To/Down end [Step steps] [Statements] Next [counter]	For n = 100 Down 0 Step 3 n = n * 2 Next n
	For loop = 0 To 20 Step 2 loop = (loop + 1) *2 Next loop

While – Wend statement

Loop an unknown number of times. Looping is controlled by the Condition. When Condition is TRUE, the statements will be executed repetitively until the condition turns to FALSE.

Condition
Necessary. An expression which evaluates to TRUE or FALSE.

Statements
Optional. Statement block.

break
Used in looping control or select statement. It skips immediately to the end of the statement.

continue
Used in looping control statement. It quits the current iteration of a loop and starts the next one.

return
Returns to the function or module that called the current function.

Format	Examples

While [*Condition*]
[Statements]
Wend
	`For n = 100 Down 0 Step 3`
	`  n = n * 2`
	`Next n`

	`For loop = 0 To 20 Step 2`
	`  loop = (loop + 1) *2`
	`Next loop`

Boolean expressions:

The value of Boolean expression is zero means FALSE.

The value of Boolean expression is not zero means TRUE.

Function calls and passing parameters

A sub-function must be defined before its execution and is therefore placed before the main() macro function. If there is an entry attempt into a function before it is first defined in the macro program will create a compiler error 'Function not defined'.

```
Sub int SQR(int x)   //This is the sub-function that squares a
number
x = x * x
return x
End Sub
Macro_Command main( ) // This main function calls the SQR()
function.
int I = 5
 I = SQR(I)
SetData(I, LW_Binplc, 1,1)  // Write the result to a LW
End Macro Command
```

The above example code defines a sub-function called SQR(). This sub-function allows one variable to be passed into it. The function then takes the value passed to it, multiplies its value by itself and then stores the result back into the same variable. The result is returned back to the calling function.

Reading & Writing External Registers in a Macro

A macro program can communicate with external PLC data registers and I/O. The function GetData() receives data from the PLC (or the HMI's local LW and LB memory). The function SetData() sends data to the PLC (or the HMI's local LW and LB memory).

The GetData() or the SetData() can be hand typed according to the format listed below, or the "PlcCom" button located at the bottom-left of the macro editor dialog box allows the user to generate the GetData() or SetData() function code by selecting the appropriate responses within the PlcCom dialog box. Be sure to declare all variables before using the 'PlcCom' option. To set up the following functions, place the cursor after the declarations, and

then press the PlcCom button. This way, all variables will be available in a drop-down menu list within the 'PlcCom' dialogHIDD MACROPLCCOM.

GetData() Function

This function reads data from a register location in the HMI or PLC memory, and stores it into a variable.

The format for the GetData() function is:

GetData(DestVariable, Addr Tpe, Address, NumElements)

Parameter	Description	Example(s)
DestVariable	The name of the variable to store the data that is captured	MyVar, InputArray[0]
Addr_Type	The device type and encoding method (binary or BCD) of the PLC data.	LW_Binplc, 4x_Binplc, 0x Binaux, RW Bcdplc
Address	The address offset in the PLC (must be written as a constant whole number – no decimals – no variables)	0, 100, (an address that has a decimal point is written here without the '.' So 1.03 is written as 103
NumElements	Number of elements to read/write (must be at least 1 and it must be a constant number) If the number of elements is more than 1, the 'DestVariable' parameter must be declared as an array.	1, 16, 100

SetData() Function:

SetData(DestData ,CString strAddr_Type , int iAddr_Off , int iDataCount **)**

Description: Writes data from a variable to a location in the PLC memory.

Parameter	Description	Example(S)
DestData	The name of the variable to retrieve the data from	
StrAddr_Type	The device type and encoding method (binary or BCD) of the PLC data	
iAddr_Off	The address offset in the PLC	
iDataCount	Number of words	

Precautions, tips & tricks when using Macros

- The size of a macro in an eob file is limited by the memory of the HMI.
- The maximum storage space of local variables in a macro is 4K bytes.
- A maximum of 256 macros are allowed in an EasyBuilder project.
- A macro may cause the HMI to lock up. Possible causes are:
- A macro contains an infinite loop with no PLC communication.
- The size of an array exceeds the storage space in a macro.
- PLC communication time may cause the macro to execute slower than expected.
- Random macro compiler errors can often be corrected by recompiling again, or canceling out of the macro editor and then reopen and compile again.
- Some macros don't work the way they should due to the declaration of some variables and how they get interpreted. Some Boolean arrays work better when defined as an array of 'short's or

‘int’s. When getting data from an external register, the variable to receive it is best declared as a ‘short’ or an array of ‘short’s.

Compiler Errors & Error Codes

When there are compile errors, the error description can be referenced by the compiler error message number.

Error message format: Macro name(: Error message number) Error Message.

- (1): "Syntax error", "identifier" - There are many possibilities for the cause of this compiler error. Simply stated, the compiler found a problem with the syntax of the statement
- (2): Used without having been initialized. - Must define the size of an array during declaration. The array size declaration must be a constant whole number, not a variable.
- (3): “Redefinition error: ” - The name of variable and function within its scope must be unique.

```
Macro Command main( )
int g[10] , g          //<- illegal – redefinition of ‘g’
For g[2] = 0 To 2
g[3] = 4
Next g[2]
End Macro_Command
```

- (4): “Function name error:” **‘identifier’** - Reserved keywords and constant can not be the name of a function

```
Macro_Command If( )   //<- illegal – used ‘If’ in the
name
```

- (5): “Statement missing” ; Statement missing “**(**“ or “**)**” - Some part of the Statement is missing, typically it is a missing parentheses

```
Macro_Command main )   //<- illegal – missing ‘(‘
```

- (6): “Missing expression in ‘If’ statement”
- (7): “Missing “Then” in If statement”
- (8:) “Missing “EndIf” ” ;
- (9:) Unfinished “If’ statement before “End If” “ ;
- (10:) “Illegal Else statement” ;

```
The format of “If” statement is:
If [logic expression]Then
[ Else [If [logic expression] Then ] ]
EndIf
Any format other than this format will cause compile
error.
```

- (11:)"There should be constant behind “Case” “ ;
- (12:)"Missing “Case” behind “Select” “ ;
- (13:)"Missing “expression” behind “Select Case” “ ;
- (14:)"Missing “End Select” statement
- (15:) Illegal “Case” statement" ;
- (16:)"Unfinished “Select” statement before “End Select” “ ;

```
The format of “Select Case” statement is:
Select Case [expression]
Case [constant]
Case [constant]
Case [constant]
Case Else
End Select
Any format other than this format will cause compile
error.
```

- (17:)" “For” statement error: missing “For” before “Next” “ ;
- (18:)"Should be integer of char variable" ;
- (19:)"Missing assign statement" ;
- (20:)"Missing keyword “To” “

- (21:)"Missing “Next” statement" ;

  The format of “For” statement is:

  ```
  For [variable] = [initial value] To [end value] [Step]
  Next [variable]
  ```

  Any format other than this format will cause compile error.
- (22:)" “While” statement error: missing “While” before “Wend” “ ;
- (23:)Missing “Wend” statement" ;

  The format of “While” statement is:

  ```
  While [logic expression]
  Wend
  ```

  Any format other than this format will cause compile error.
- (24:) “Illegal “Break” statement” ;

  “Break” statement can only be used in “For”, “While”, or “Select Case” statement
  “Break” statement takes one line of Macro.
- (25:)"Illegal “Continue” statement" ;

  “Continue” statement can only be used in “For” statement, or “While” statement
  “Continue” statement takes one line of Macro.
- (26:)"Expression error" ;
- (27:) Illegal operation object" ;

  The mismatch of operation object in expression cause compile error.
  For example :

  ```
  Macro_Command main( )
  int g[10]
  For g[2] = 0 To 2
  g[3] = 4 + XYZ  <- illegal – XYZ is undefined
  Next g[2]
  End Macro_Command
  ```
- (28:)"Missing “Sub” “ ;
- (29:)"Missing “Macro_Command” “ ;

  The format of function declaration is:

  ```
  Sub(Macro_Command) [data type] function_name( … )
  ```

  End Sub(Macro_Command)
  Any format other than this format will cause compile error.
- (30:)"Mismatch of the number of parameters “ ;
- (31:)"Mismatch of data type of parameter" ;
- (32:)"Parameter error";

  The parameters of a function must be equivalent to the arguments passing to a function to avoid compile error.
- (33:) “Undefined function” ;
- (34:)Illegal member of array
- (35:)Illegal definition of array";
- (36)Illegal index of array
- (37:) “Undefined symbol” ;

  Any variable or function should be declared before use.
- (38:) “Un-supported PLC data address” ;

  The parameter of GetData(…) , SetData(…) should be legal PLC address.
- (39:) Should be integer, character or constant" ;

  The format of array is:
  Declaration: array_name[constant] (constant is the size of the array)
  Usage: array_name[integer, character or constant] Any format other than this format will cause compile error.
- (40:) “Illegal Macro statement before declaration statement ”

  For example :

  ```
  Macro_Command main( )
  int g[10]
  ```

```
For g[2] = 0 To 2
g[3] = 4 + g[9]
int h , k <- illegal – definitions must occur
before any statements or expressions
Next g[2]
End Macro_Command
```

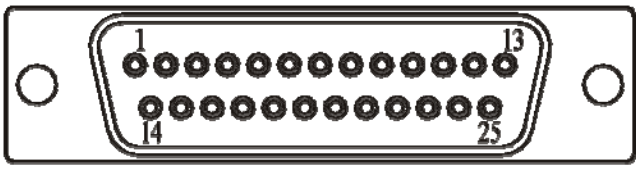
- (41:) “Floating point can not bitwise shift”
- (42:) “Missing function return value ” ;
- (43) “Function can not return a value”
- (44:) “Illegal Float data type in expression” ;
- (45:) “Error PLC address” ;
- (46:) “Stack can not exceed 4k bytes” ;
- (47:) “Only one main entrance in the Macro is allowed” ;
- (48) “Too many main entrance: ” **‘identifier’**

  The only one main entrance of Macro is:

```
Macro Command function name( )
End Macro Command
```

Chapter 14 - Using a Printer With the MT508/550

The MT508/550 models support Epson ESC/P2, SP, and HP PCL/Simple Page Mode printers. The MT508/550 provides a standard 25-pin female connector for connection to the printer with the following pin configuration.



DB25S Female, Solder Cup Side

Printer pinout (25p D-Female)

Pin	Description	In/Out
1	STB	OUT
2	DATA0	OUT
3	DATA1	OUT
4	DATA2	OUT
5	DATA3	OUT
6	DATA4	OUT
7	DATA5	OUT
8	DATA6	OUT
9	DATA7	OUT
11	BUSY	OUT
15	ERROR	IN
16	INIT	OUT
17-25	GND	–

The MT506 Models do not support a printer.

Supported Printer Models

Below is a partial list of printer models that are supported by the MT5xx' printer protocol. Models with an asterisk (*) next to it have been tested by Weintek, Inc.

- **Epson** LQ-570e*, 680Pro, 870
- **Panasonic** KX-P1131, P3123, P3124
- **Epson Stylus** 300, 400, 480SXU, 800, 800+, 1000, 1500
- **Epson Stylus Color** 200, 400, 500, 600*, 740, 800, 950, 900, 950, 980, 980N
- **Epson Stylus Pro** ProXL
- **Epson Stylus Photo** 750
- **HP DeskJet** 450, 520*, 640C, 656C, 845C, 920C, 930C, 1125C, 1220C*, 5550, 6122

Most HP inkjets and small laser jets work well. It is possible that other printers work with the MT5xx; however, they have not been tested.

Non-supported Printer Models

The following printer models are not supported by the MT5xx:

- Any **Epson** 9-pin dot-matrix
- **Epson** Stylus C40UX, C60, C80
- **Epson** FX-850, 880+, 980
- **Epson** LQ200, 400, 450, 500, 550, 850, 860
- **Epson** LX-300, 300+
- **Epson** SQ-850
- **Okidata Microline** 184, 395
- **Okidata Pacemark** - all models

Configuring the MT508/550 for a Printer

1. From EasyBuilder's Edit menu, select **System Parameters**.
2. Select the **General** tab.
3. In the Print area (near the bottom of the dialog box), set the Printer to the desired printer. If **None** is selected, no printing will occur.
4. If each printed Event is to have a unique serial number printed with it, check the **Print Sequence Number** checkbox.
5. If each printed Event is to have the current time printed with it, check the **Print Time Tag** checkbox.
6. In the drop-down list, select how the MT508/550 is to handle printer errors:
   - No error detection: Printer errors are not detected, and data is lost.
   - Error detection (show message): The HMI displays a printer error message, and the data to print is buffered until the printer is ready.
   - Error detection (set local bit): The HMI sets local bit LB9016, and the data to print is buffered until the printer is ready. Bit LB9016 is cleared when the printer is ready.
   - The Sequence Number and Time Tag options apply only to Event data. Printed Windows will not have a Sequence Number or Time Tag printed with them.

The MT508/550 can send the following to the printer:

- (see Chapter 9)
- (see below)
- Reports (see below)

Printing a Window

There are two ways to print a window:

- Using a function key
- Using a PLC Control Object
- Using Simulation Mode

Using a Function Key

You can place a Function Key Object on a screen that is configured to print the screen.

When the Function Key is touched, the Window containing the Function Key is sent to the printer.

▶ **To create a Function Key for printing a Window, follow these steps:**

1. From the **Parts** menu, select **Function Key**.

2. Under the **General** tab, select the **Hard Copy** option button. The **Attributes** button is enabled.
3. Select the **Attributes** button. The Attributes determine what part(s) of the Window are printed.
   - **Print Text & Meter & Trend**
     The printed Window contains only the window's Text, Meter, & Trend objects.
   - **Print Text & Meter & Trend & All Shape but not including Pattern**
     The printed Window contains only the window's Text, Meter, Trend, and Shape objects. All shapes on the Window are printed, whether they are part of another object (Bit Lamp, Function Key, etc.) or stand-alone Shape objects. However, the Patterns associated with the shapes are not printed.
   - **Print Text & Meter & Trend & All Bitmap**
     The printed Window contains only the window's Text, Meter, Trend, and Bitmap objects. All bitmaps on the Window are printed, whether they are part of another object (Bit Lamp, Function Key, etc.) or stand-alone Bitmap objects.
   - **Print Text & Meter & Trend & All Bitmap & All Shape but not including Pattern**
     The printed Window contains only the window's Text, Meter, Trend, Bitmap, and Shape objects. All shapes and bitmaps on the Window are printed, whether they are part of another object (Bit Lamp, Function Key, etc.) or stand-alone Shape and Bitmap objects. However, the Patterns associated with the shapes are not printed.
   - **Print Text & Meter & Trend & All Bitmap & All Shape**
     The printed Window contains only the window's Text, Meter, Trend, Bitmap, and Shape objects. All shapes and bitmaps on the Window are printed, whether they are part of another object (Bit Lamp, Function Key, etc.) or stand-alone Shape and Bitmap objects.
   - **Next Page**
     Advances the Printer to the next page.

If desired, attach a Shape and a Label to the Function Key.

Using PLC Control Object

You can print a copy of the screen that is currently displayed on the HMI by using the PLC Control Object. This feature allows you to print a hardcopy of the current screen whenever directed by the PLC.

Using the PLC Control Object to print the HMI screen

1. From the **Parts** menu, select **PLC Control**. The PLC Control Object dialog box appears.
2. Click **Add**. The PLC Control Object's Attribute dialog box appears.
3. Select **Screen hardcopy** for **Type of control** in the Attribute box.
4. Select the **Device type** and **Device Address** for the actual PLC bit address you wish to use.
5. Click **OK** to return to the PLC Control Object dialog box.

You should see a new entry, which lists the PLC address that is monitored by the HMI. The PLC can now signal the HMI to print the screen currently displayed. The HMI will send the screen contents to a printer when the PLC sets the specified PLC bit.

Using Simulation Mode

Printing a window can also be done during Simulation (Online & Offline).

To Print a window:

1. Place the mouse anywhere on the simulation screen.
2. Right click the mouse to call up the menu.
3. Select the desired menu item.
   - Print screen (preview) & (save to file): Prints a picture of the current screen being displayed. To preview before printing, select **Screen Preview. Print screen to a file** will create a bitmap image of the screen.

- Print window (preview) & (save to file): To print object/address information within a window, select **Window printing** and select the **Table** radio button. To preview, select **Window pre- view. Print window to a file** creates a bitmap image of the screen data.

Printing a Report

You can print a report by using the PLC Control Object. This feature allows you to print a report that is preconfigured in the HMI whenever directed by the PLC.

Using the PLC Control Object to print a report

1. From the **Parts** menu, select **PLC Control**. The PLC Control Object dialog box appears.
2. Click **Add**. The PLC Control Object's Attribute dialog box appears.
3. Select Report printout for Type of control: in the Attribute box.
4. Select the **Device type** and **Device Address** for the actual PLC address you wish to monitor.
5. Select either **BIN** (binary) or **BCD** format.
6. Click **OK** to return to the PLC Control Object dialog box.
7. You should see a new entry, which lists the PLC address that is monitored by the HMI.

The PLC can now signal the HMI to print a report. A report is a pre-configured screen in the HMI. When the HMI reads the PLC Control Object: Report Printout PLC address, it will do the following:

- Determine if the screen requested is valid
- If valid, it will update all data registers that are on the screen
- Then the HMI will send a copy of the screen to the printer.

The screen does not have to be showing on the HMI display.

Appendix A - Specifications

MT506

Mechanical

Material: Plastic PBT & PC housing with polyester overlay & neoprene gasket

Mounting: Panel, 1/8 inch [3.2mm] nominal thickness

Wiring: Two 9-position shielded D-sub serial communications connectors plus power connector

Weight: 1.8 pounds [0.8 kg]

Environmental

Protection: Sealed to NEMA 4/12 (IP65) when properly panel mounted (O-ring seal)

Operating Temperature: +32 to +113º F [0 to 45º C]

Storage Temperature: -4 to +140º F [-20 to 60º C]

Operating Humidity: 10-90% RH (non-condensing)

Electrical Noise Emissions and Immunity

Emissions:EN55011 (Group 1, Class B)—Generic commercial, light & heavy industrial environments

EN50081-1 Generic domestic and light industrial environments

EN50081-2 Generic heavy industrial environments

Immunity:EN50082-1 Generic domestic and light industrial environments

EN50082-2 Generic heavy industrial environments

Power Requirements

Input Voltage: 24 VDC 5%

Input Current: 1 amp instantaneous, 400 milliamps continuous

Display

Type: Liquid Crystal Display (LCD), 320 x 240 pixels or 240 x 320 (MT506T 320 x 234 or 234 x 320)

Size: 5.7 inches [145mm] diagonal

Backlight: CCFL with 15,000 hour minimum lifespan (mono version)
25,000 hour minimum lifespan (color version)

MT506M: 4-shade blue-mode STN, 60 cd/m², 15:1 contrast

MT506C: 256-color STN, 150 cd/m², 30:1 contrast

MT506H: 256-color TFT, 300 cdm^2, 100:1 contrast

MT506T: 256-color TFT, 500 cdm^2.150:1 contrast, 320 x 234 pixels

Touchscreen

Type: 4-wire analog resistive

Touch Accuracy: 0.04 inches [1.5 mm] resolution

Surface Hardness: 4H6

Communications

Serial ports:Port 1 is RS-232 communications to PLC (DE9S)

Port 2 is RS-485 communications to PLC and RS-232 PC configuration (DE9P)

Baud Rates:9600, 19200, 38400, 57600, 115200

Type:Point-to-point serial communications

Internal Features

Microprocessor: 200 MHz Intel XScale

Recipe and real-time clock module

Memory: 4MB DRAM, 1MB flash, 128 KB recipe

MT508

Mechanical

Material: Plastic PBT & PC housing with polyester overlay & neoprene gasket

Mounting: Panel, 1/8 inch [3.2mm] nominal thickness

Wiring: Two 9-position shielded D-sub serial communications connectors plus power connector and standard printer cable connector

Weight: 2.64 pounds [1.2 kg]

Environmental

Protection: Sealed to NEMA 4/12 (IP65) when properly panel mounted (O-ring seal)

Operating Temperature: +32 to +113º F [0 to 45º C]

Storage Temperature: -4 to +140º F [-20 to 60º C]

Operating Humidity: 10-90% RH (non-condensing)

Electrical Noise Emissions and Immunity

Emissions:EN55011 (Group 1, Class B)—Generic commercial, light & heavy industrial environments

EN50081-1 Generic domestic and light industrial environments

EN50081-2 Generic heavy industrial environments

Immunity:EN50082-1 Generic domestic and light industrial environments

EN50082-2 Generic heavy industrial environments

Power Requirements

Input Voltage: 24 VDC 5%

Input Current: 1.5 amps instantaneous, 425 milliamps continuous

Display

Type: Liquid Crystal Display (LCD), 640 x 480 pixels or 480 x 640

Size: 7.7 inches [196mm] diagonal

Backlight: CCFL x 2 with 25,000 hour minimum lifespan

MT508C: 256-color DSTN, 150 cd/m², 30:1 contrast

MT508T: 256-color TFT, 400 cdm ^{2}, 250:1 contrast

Touchscreen

Type: 4-wire analog resistive

Touch Accuracy: 0.08 inches [2mm] resolution

Surface Hardness: 4H

Communications

Serial ports:Port 1 is RS-232 communications to PLC (DE9S)

Port 2 is RS-485 communications to PLC and RS-232 PC configuration (DE9P)

Parallel port:Standard parallel printer port

Baud Rates:9600, 19200, 38400, 57600, 115200

Type:Point-to-point serial communications

Internal Features

Microprocessor: 200 MHz Intel XScale

Recipe and real-time clock module

Memory: 4MB DRAM, 1MB flash, 128KB recipe

MT510

Mechanical

Material: Plastic PBT & PB housing with polyester overlay & neoprene gasket

Mounting: Panel, 1/8 inch [3.2mm] nominal thickness

Wiring: Two 9-position shielded D-sub serial communications connectors plus power connector and standard printer cable

Weight: 4.4 pounds [2 kg]

Environmental

Protection: Sealed to NEMA 4/12 (IP65) when properly panel mounted (O-ring seal)

Operating Temperature: +32 to +113º F [0 to 45º C]

Storage Temperature: -4 to +140º F [-20 to 60º C]

Operating Humidity: 10-90% RH (non-condensing)

Electrical Noise Emissions and Immunity

Emissions*:*

EN55011 (Group 1, Class B)—Generic commercial, light & heavy industrial environments

EN50081-1 Generic domestic and light industrial environments

EN50081-2 Generic heavy industrial environments

Immunity*:*

EN50082-1 Generic domestic and light industrial environments

EN50082-2 Generic heavy industrial environments

Power Requirements

Input Voltage: 24 VDC 5%

Input Current: 2 amps instantaneous, 500 milliamps continuous

Display

Type: Liquid Crystal Display (LCD), 640 x 480 pixels or 480 x 640

Size: 10.4 inches [264mm] diagonal

Backlight: CCFL x 2 with 25,000 hour minimum lifespan

MT510H: 256-color TFT, 250 cd/m², 100:1 contrast *MT510M*:

4-shades of gray DSTN, 150 cd/m², 30:1 contrast

Touchscreen

Type: 8-wire analog resistive

Touch Accuracy: 0.08 inches [2mm] resolution

Surface Hardness: 4H

Communications

Serial ports:

Port 1 is RS-232 communications to PLC (DE9S)

Port 2 is RS-485 communications to PLC and RS-232 PC configuration (DE9P)

Parallel port:

Standard parallel printer port

Baud Rates:

9600, 19200, 38400, 57600, 115200

Type:

Point-to-point serial communications

Internal Features

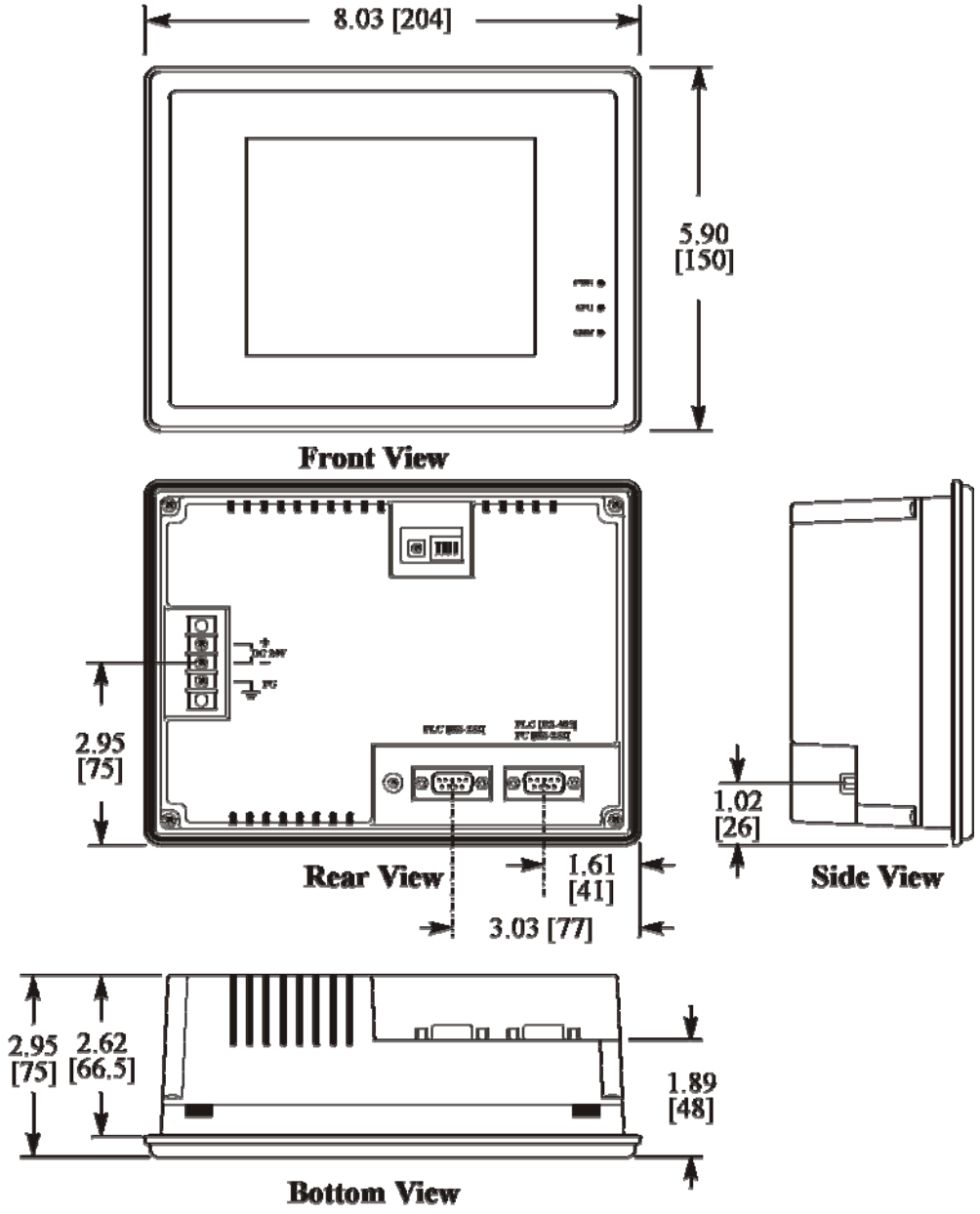
Microprocessor: 200 MHz Intel XScale

Recipe and real-time clock module

Memory: 4MB DRAM, 1MB flash, 128 KB recipe

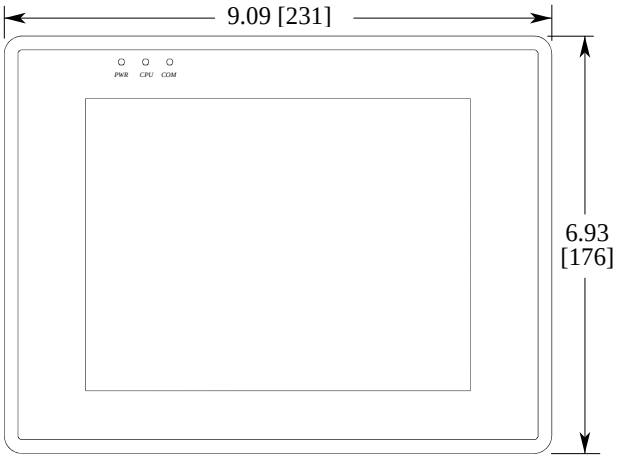
Appendix B - Dimensional Outlines

MT506

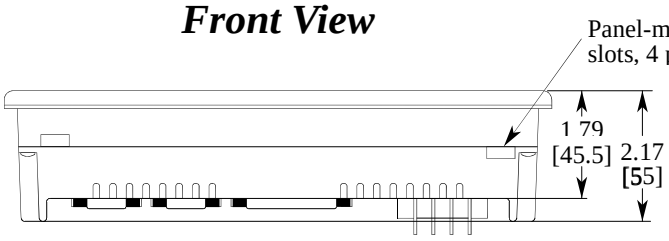


Dimensions are in inches [mm]

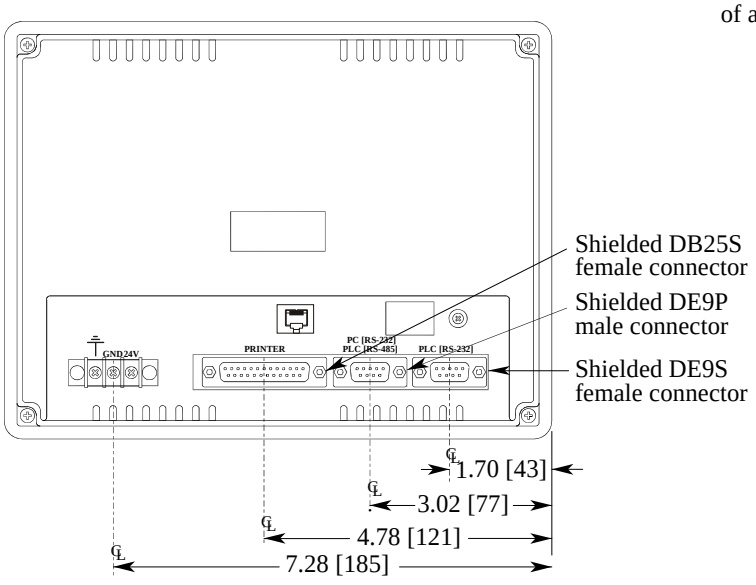
MT508



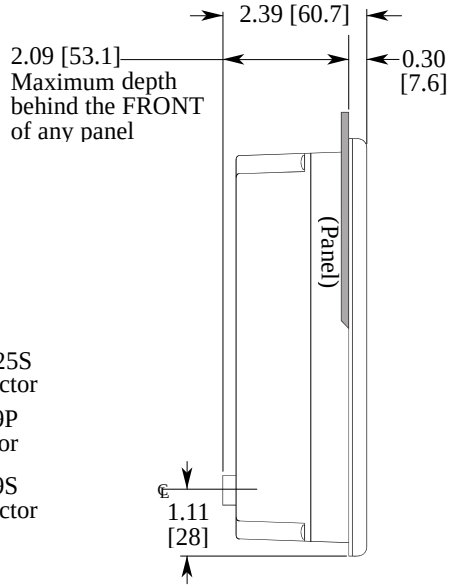
Front View



Bottom View



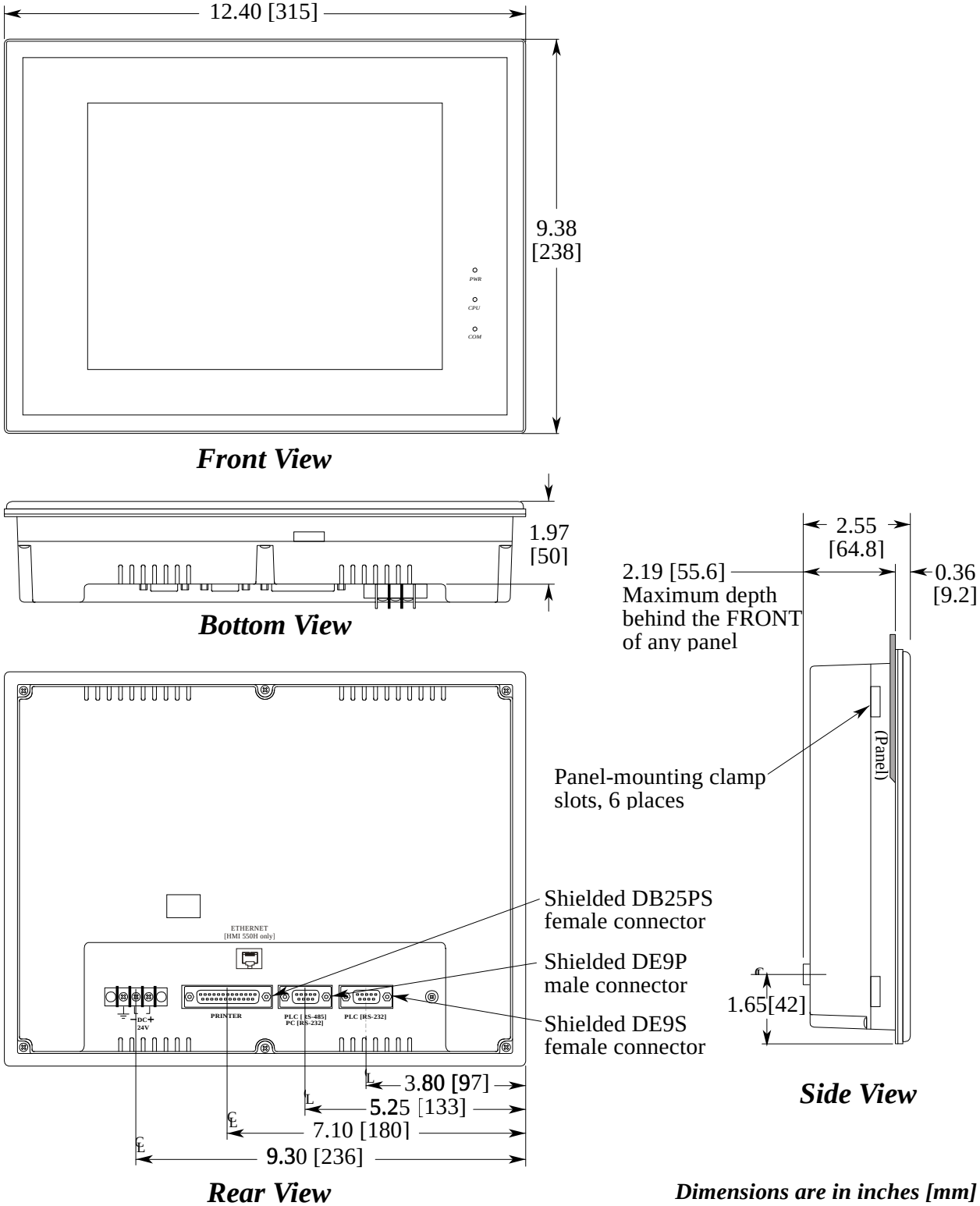
Rear View



Side View

Dimensions are in inches [mm]

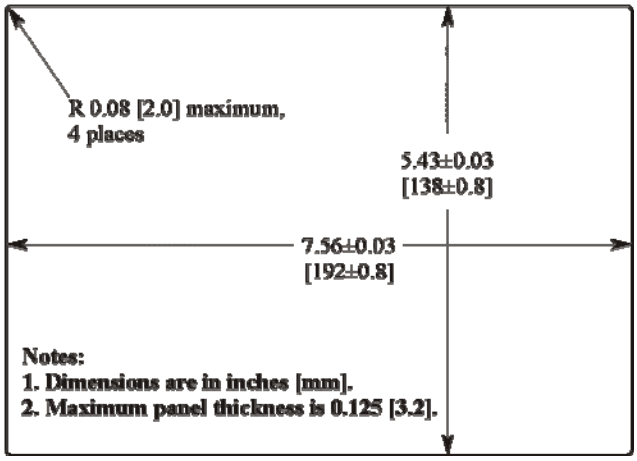
MT510



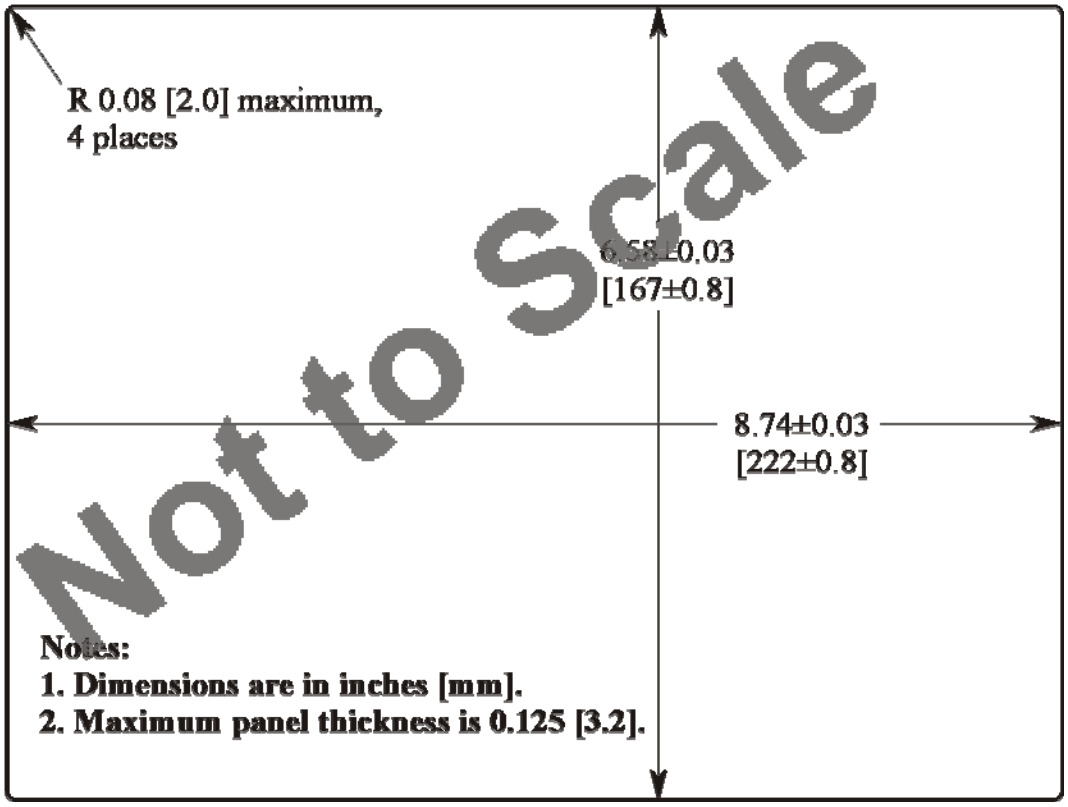
Dimensions are in inches [mm]

Appendix C - Panel Cutout Dimensions

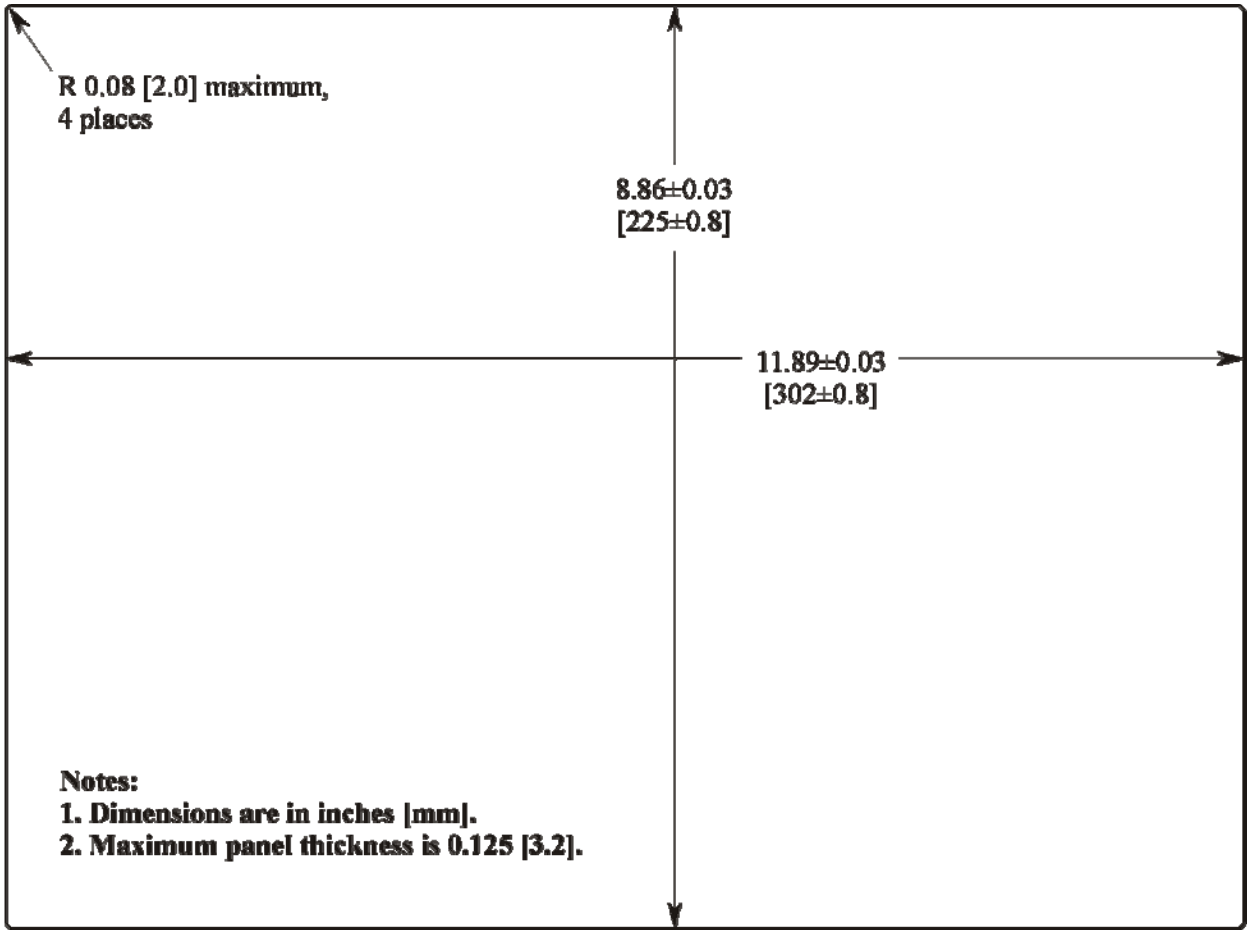
MT506



MT508

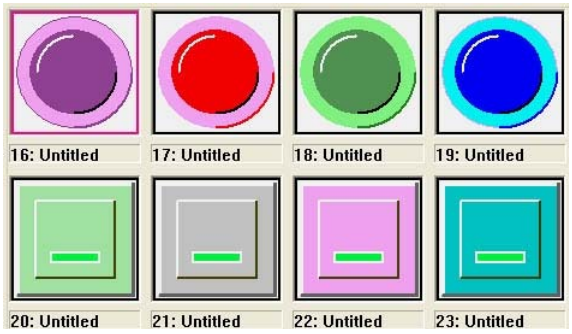


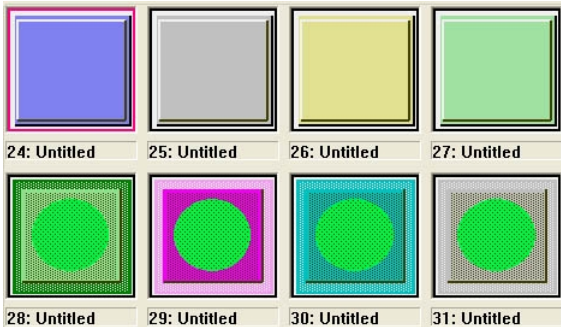
MT510

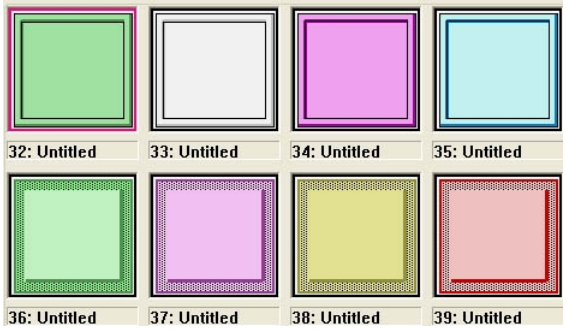

R 0.08 [2.0] maximum,
4 places
8.86±0.03
[225±0.8]
11.89±0.03
[302±0.8]
Notes:
1. Dimensions are in inches [mm].
2. Maximum panel thickness is 0.125 [3.2].

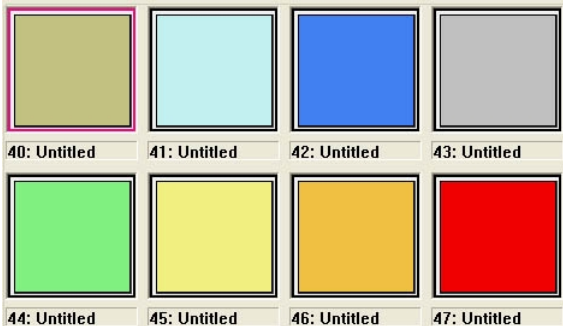
Appendix D - Libraries

button1.slb

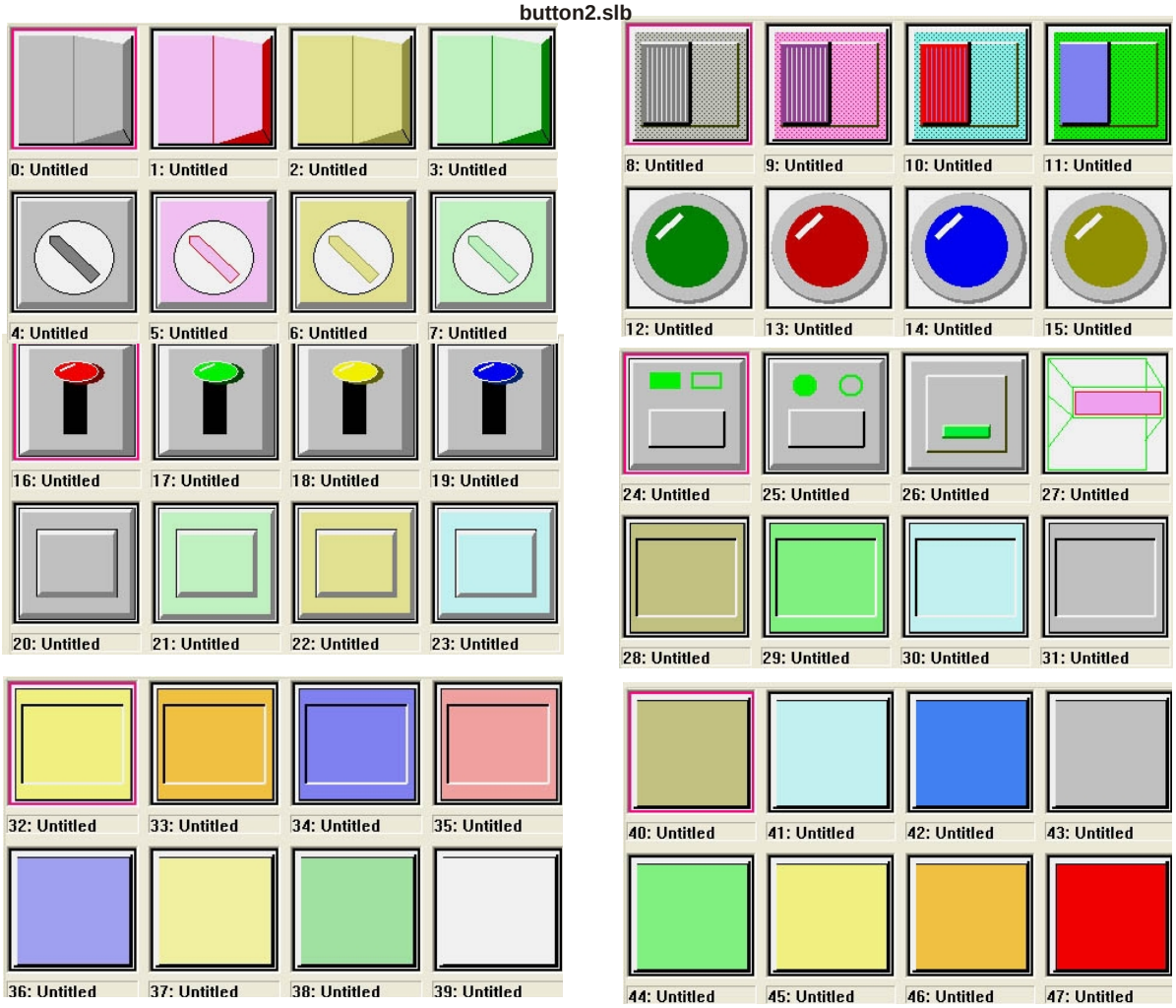




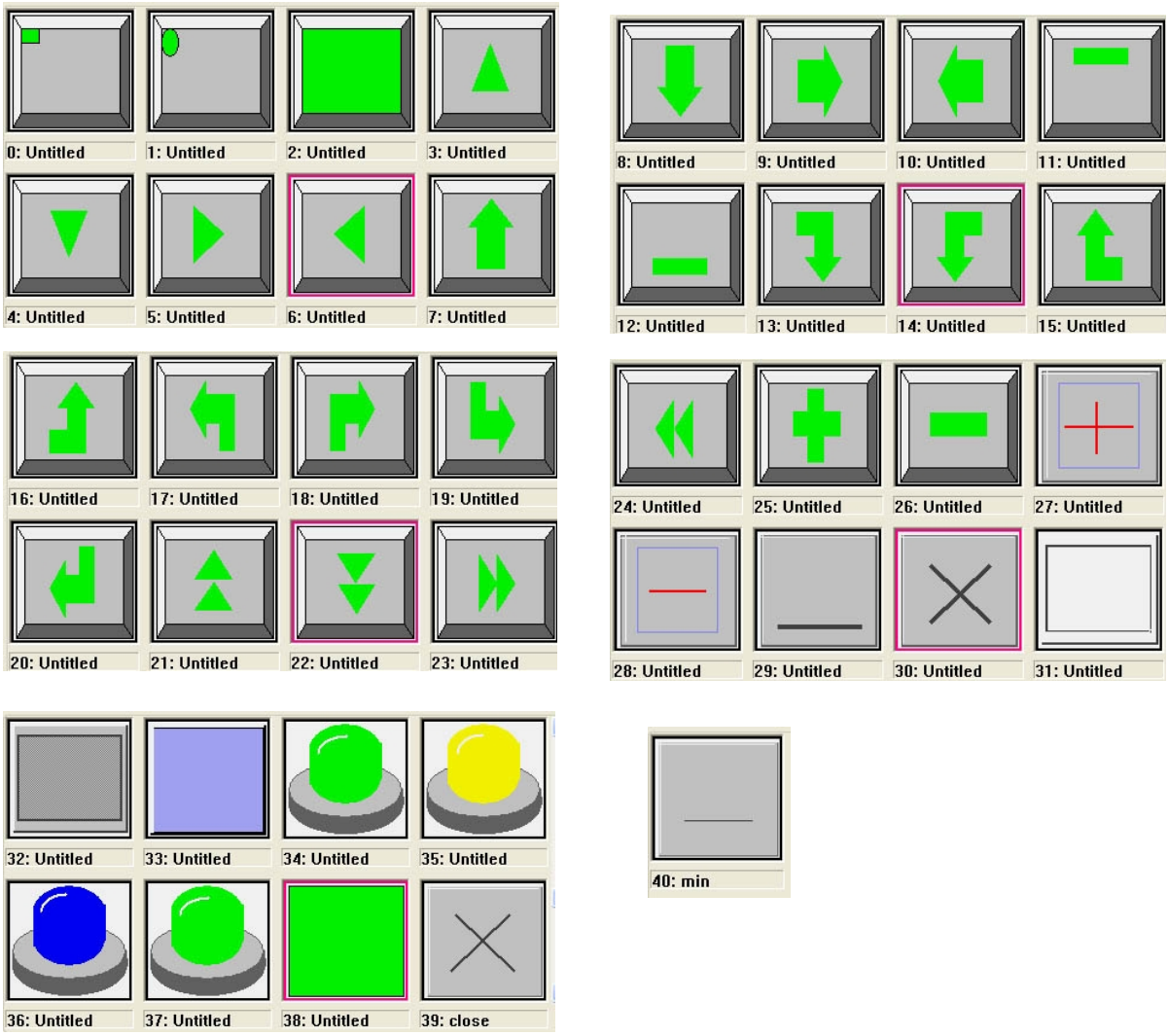




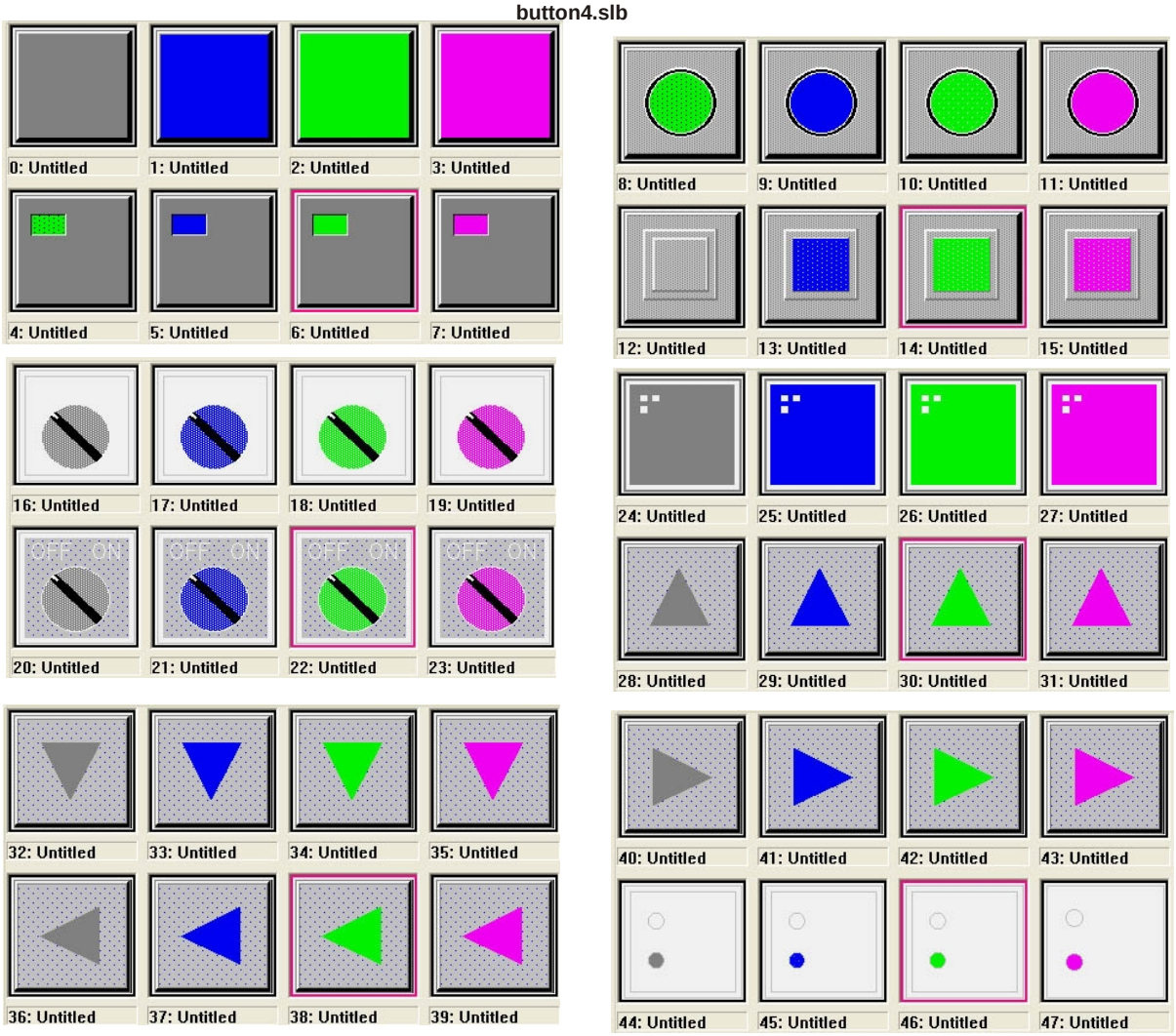
button2.slb


0: Untitled
1: Untitled
2: Untitled
3: Untitled
4: Untitled
5: Untitled
6: Untitled
7: Untitled
8: Untitled
9: Untitled
10: Untitled
11: Untitled
12: Untitled
13: Untitled
14: Untitled
15: Untitled
16: Untitled
17: Untitled
18: Untitled
19: Untitled
20: Untitled
21: Untitled
22: Untitled
23: Untitled
24: Untitled
25: Untitled
26: Untitled
27: Untitled
28: Untitled
29: Untitled
30: Untitled
31: Untitled
32: Untitled
33: Untitled
34: Untitled
35: Untitled
36: Untitled
37: Untitled
38: Untitled
39: Untitled
40: Untitled
41: Untitled
42: Untitled
43: Untitled
44: Untitled
45: Untitled
46: Untitled
47: Untitled

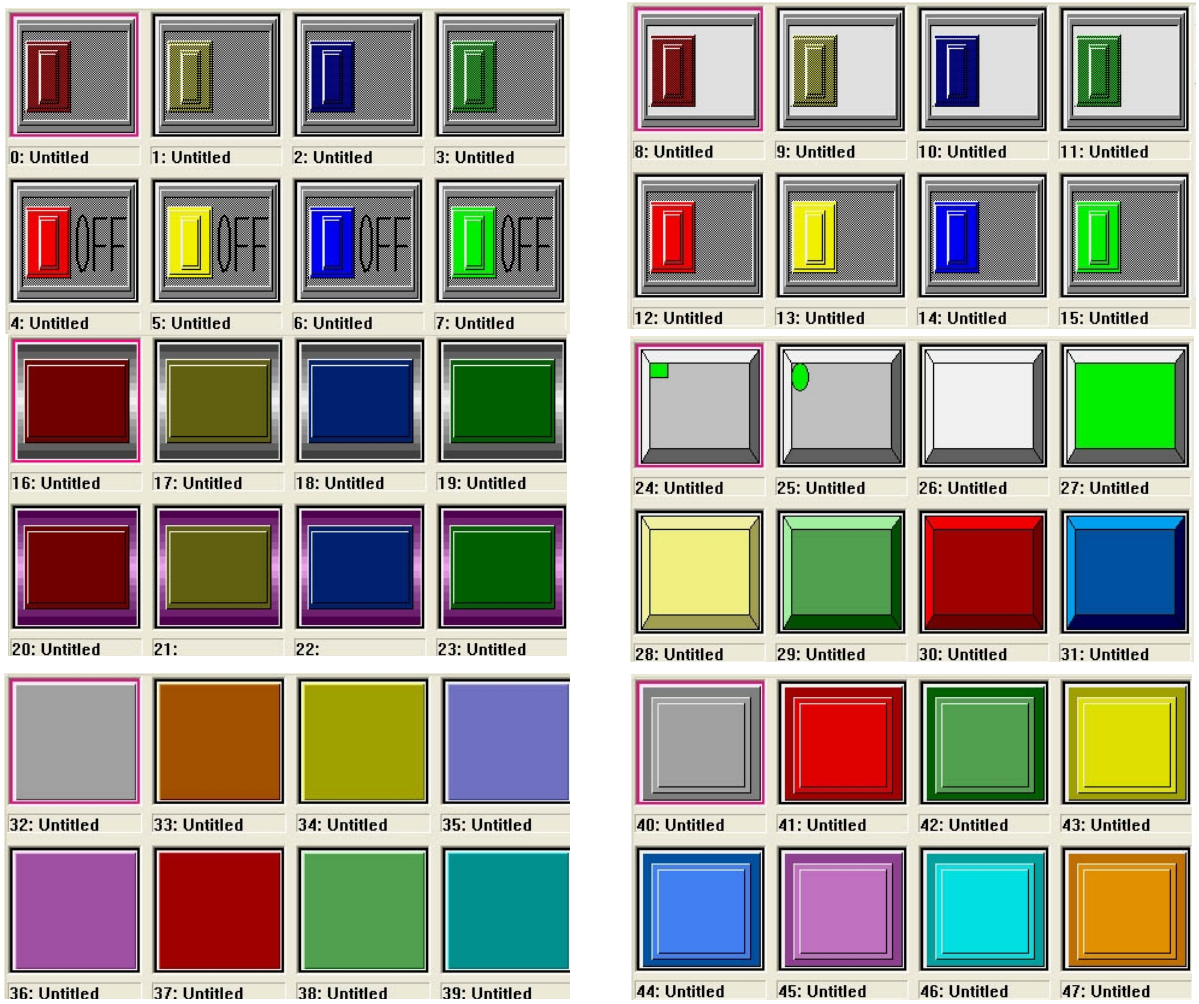
button3.slb


0: Untitled
1: Untitled
2: Untitled
3: Untitled
4: Untitled
5: Untitled
6: Untitled
7: Untitled
8: Untitled
9: Untitled
10: Untitled
11: Untitled
12: Untitled
13: Untitled
14: Untitled
15: Untitled
16: Untitled
17: Untitled
18: Untitled
19: Untitled
20: Untitled
21: Untitled
22: Untitled
23: Untitled
24: Untitled
25: Untitled
26: Untitled
27: Untitled
28: Untitled
29: Untitled
30: Untitled
31: Untitled
32: Untitled
33: Untitled
34: Untitled
35: Untitled
36: Untitled
37: Untitled
38: Untitled
39: close
40: min

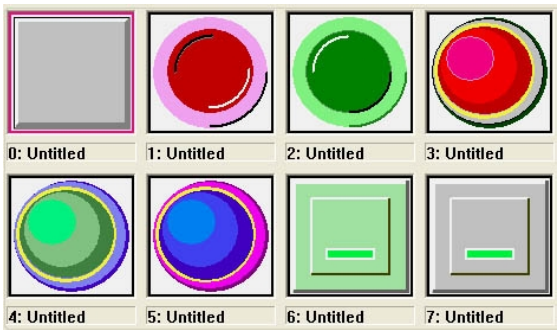
button4.slb

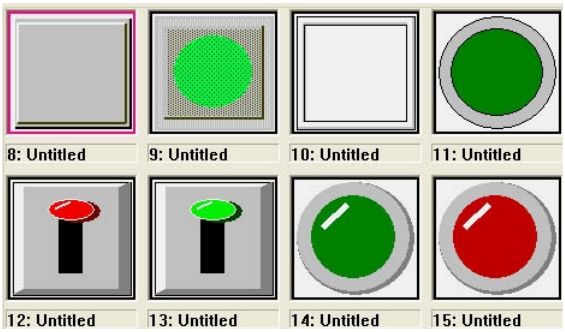

0: Untitled
1: Untitled
2: Untitled
3: Untitled
4: Untitled
5: Untitled
6: Untitled
7: Untitled
8: Untitled
9: Untitled
10: Untitled
11: Untitled
12: Untitled
13: Untitled
14: Untitled
15: Untitled
16: Untitled
17: Untitled
18: Untitled
19: Untitled
20: Untitled
21: Untitled
22: Untitled
23: Untitled
24: Untitled
25: Untitled
26: Untitled
27: Untitled
28: Untitled
29: Untitled
30: Untitled
31: Untitled
32: Untitled
33: Untitled
34: Untitled
35: Untitled
36: Untitled
37: Untitled
38: Untitled
39: Untitled
40: Untitled
41: Untitled
42: Untitled
43: Untitled
44: Untitled
45: Untitled
46: Untitled
47: Untitled

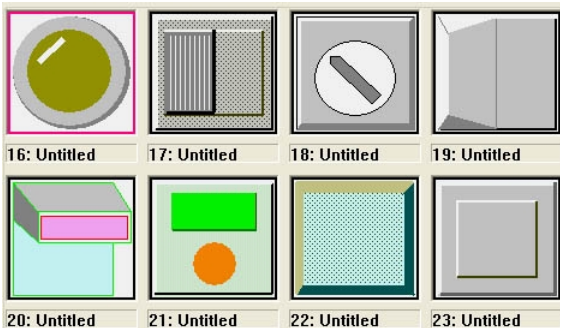
user-shape1.slb

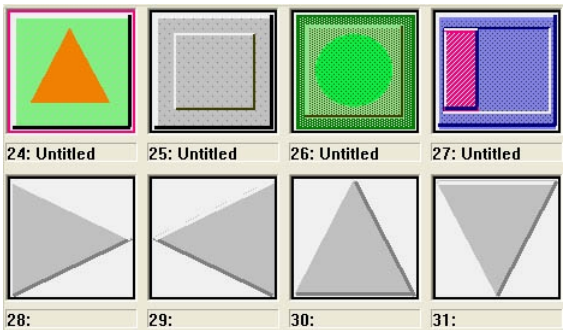


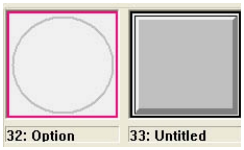
Buttons.slb


0: Untitled
1: Untitled
2: Untitled
3: Untitled
4: Untitled
5: Untitled
6: Untitled
7: Untitled

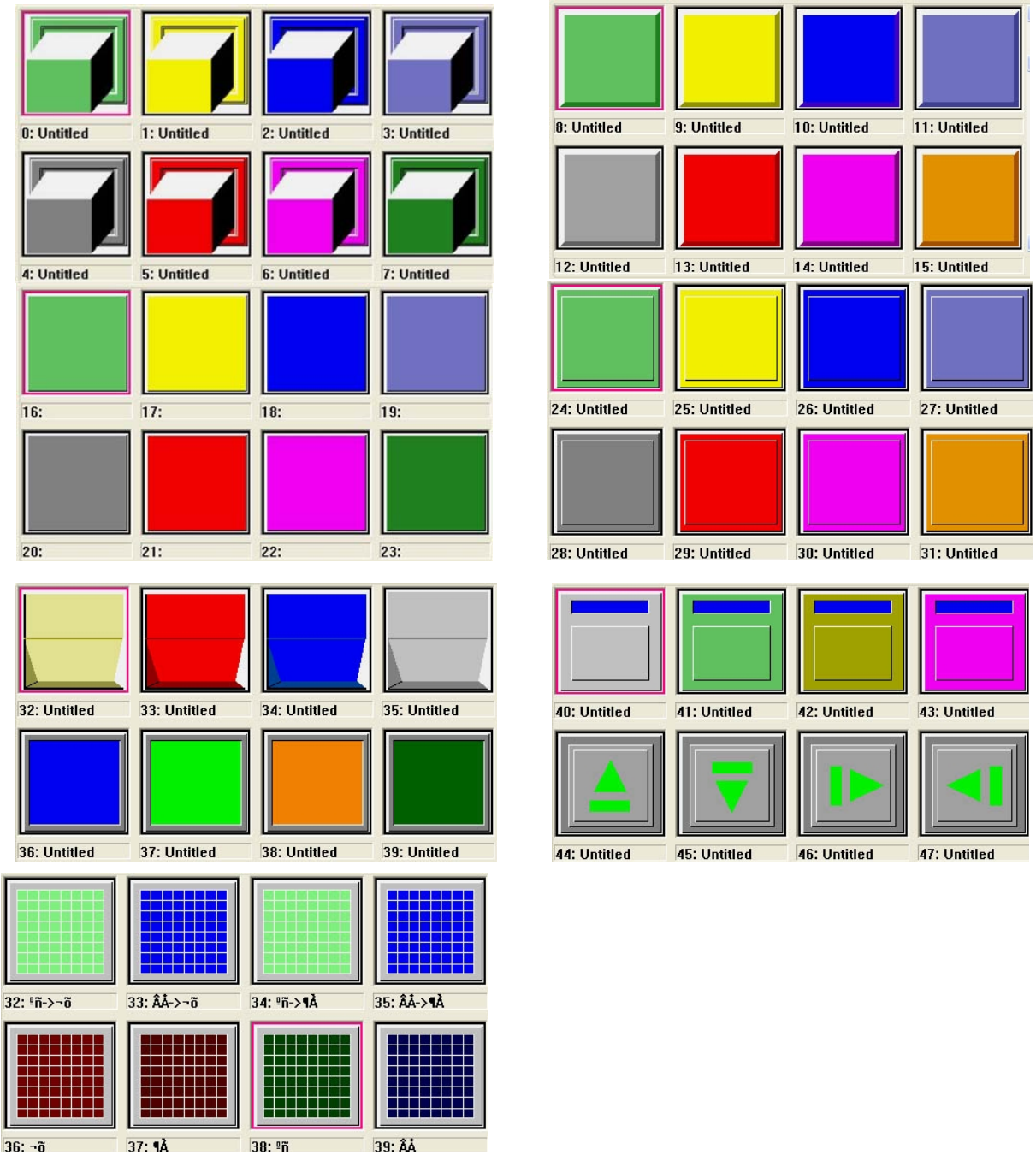

8: Untitled
9: Untitled
10: Untitled
11: Untitled
12: Untitled
13: Untitled
14: Untitled
15: Untitled


16: Untitled
17: Untitled
18: Untitled
19: Untitled
20: Untitled
21: Untitled
22: Untitled
23: Untitled


24: Untitled
25: Untitled
26: Untitled
27: Untitled
28:
29:
30:
31:

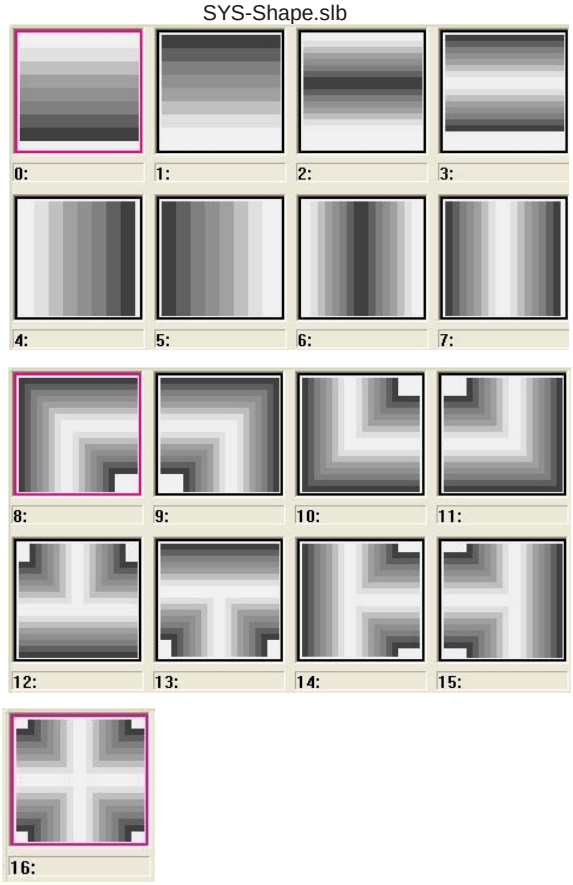

32: Option
33: Untitled

user-shape.slb

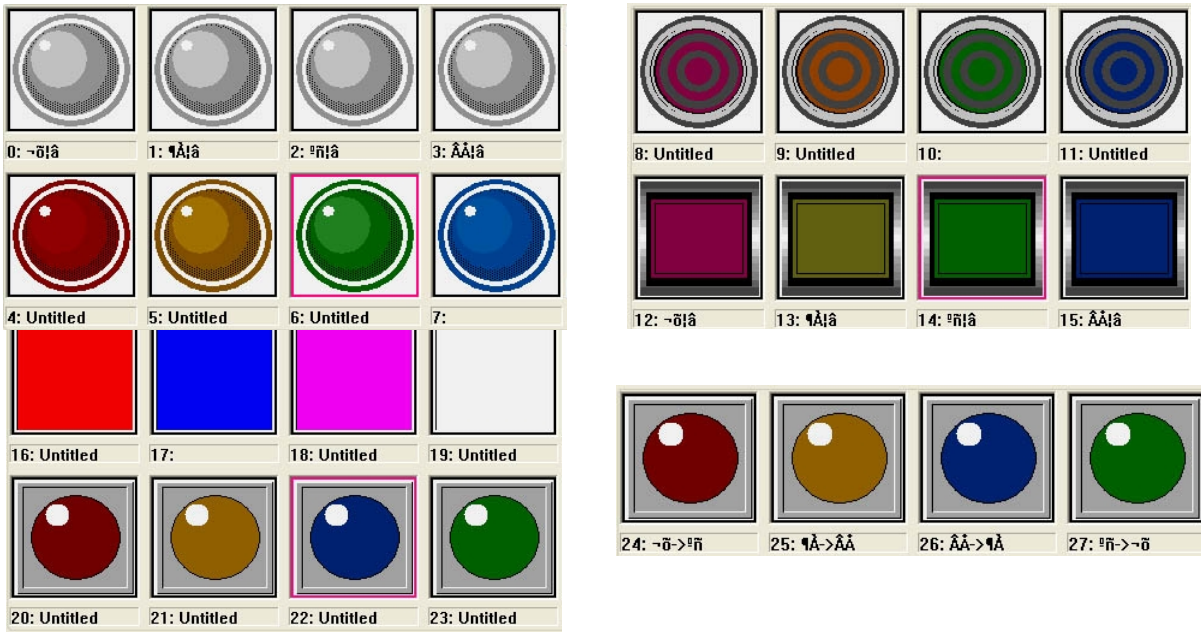


SYS_Button.slb

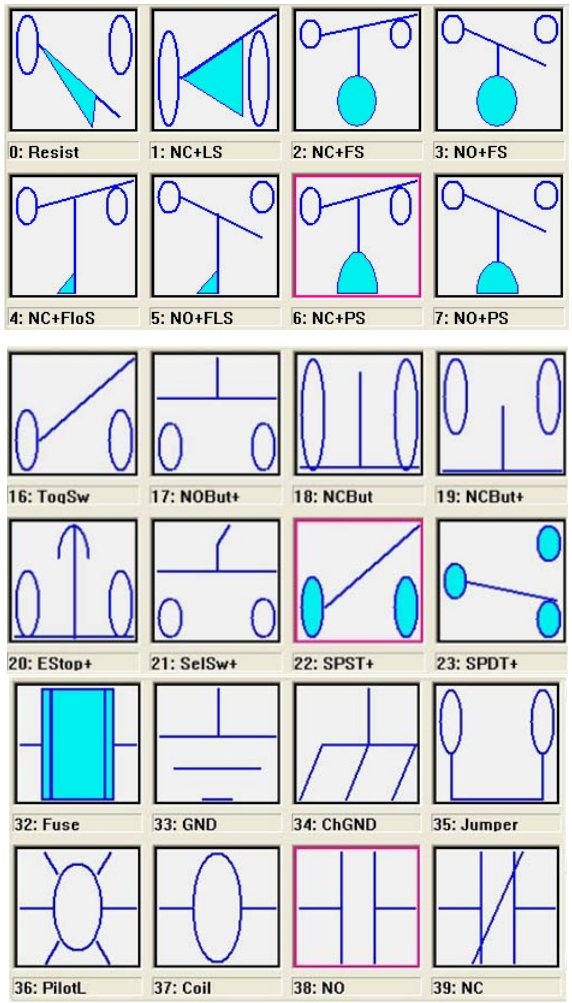
SYS-Shape.slb

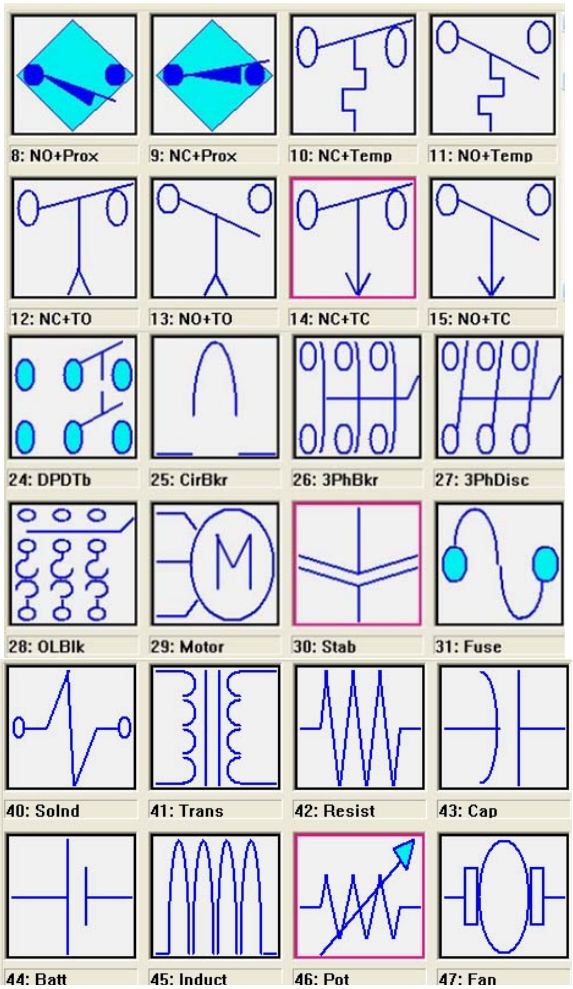


user-shape-lamp.slb

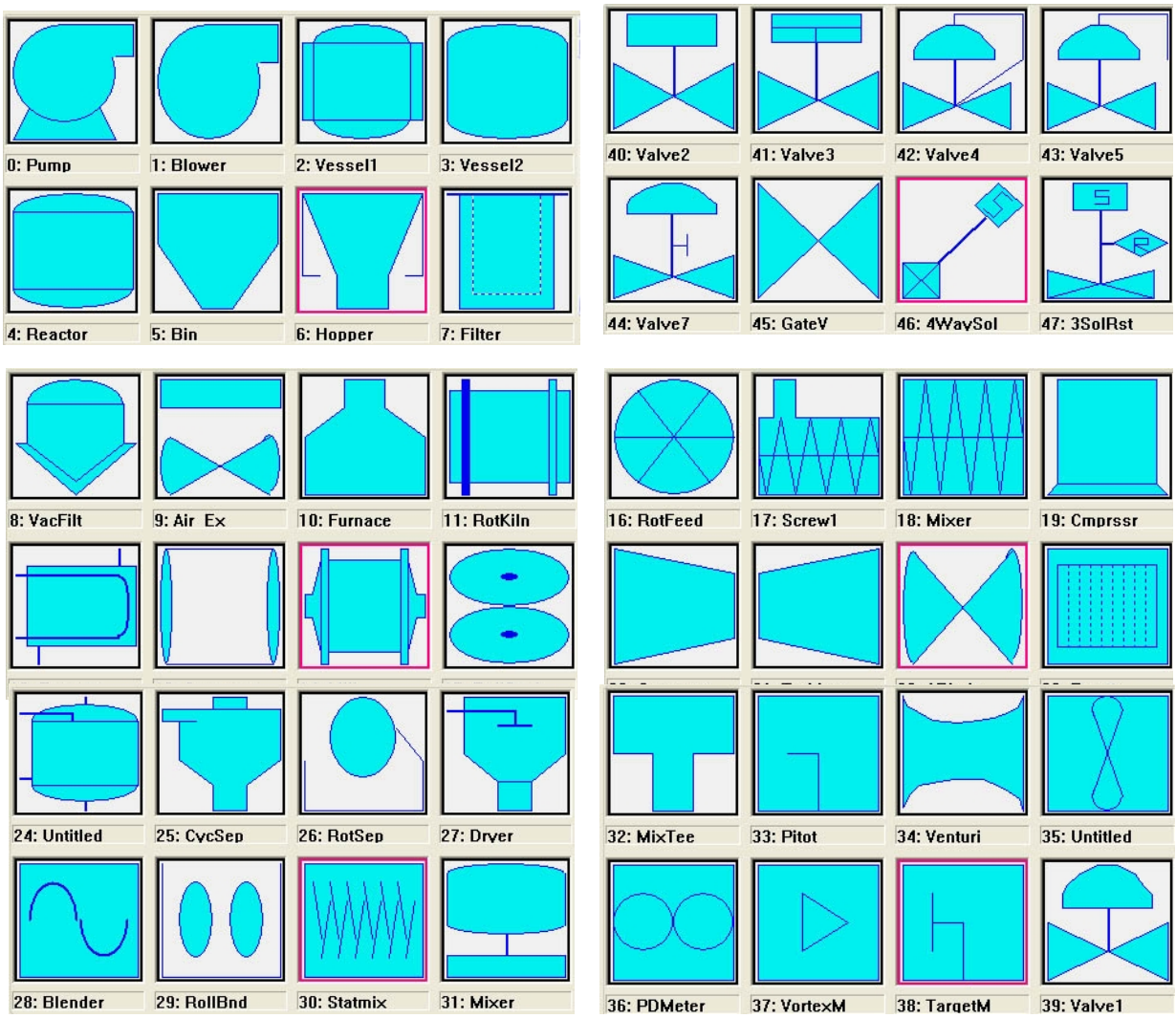


ELECTRIC.slb

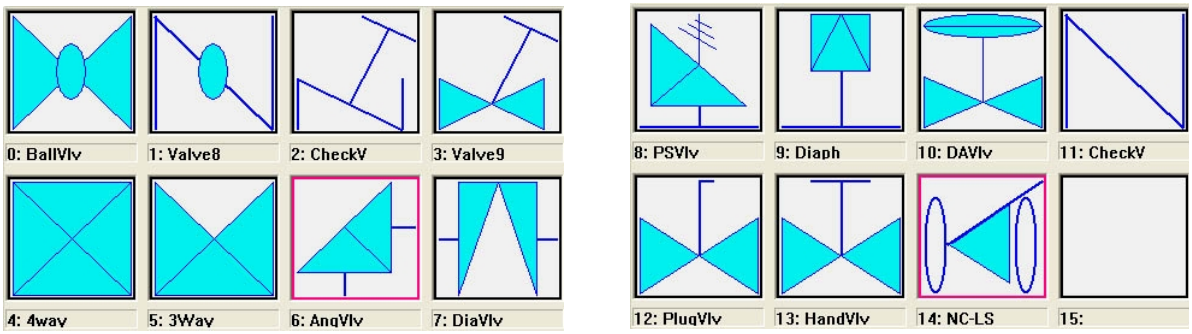




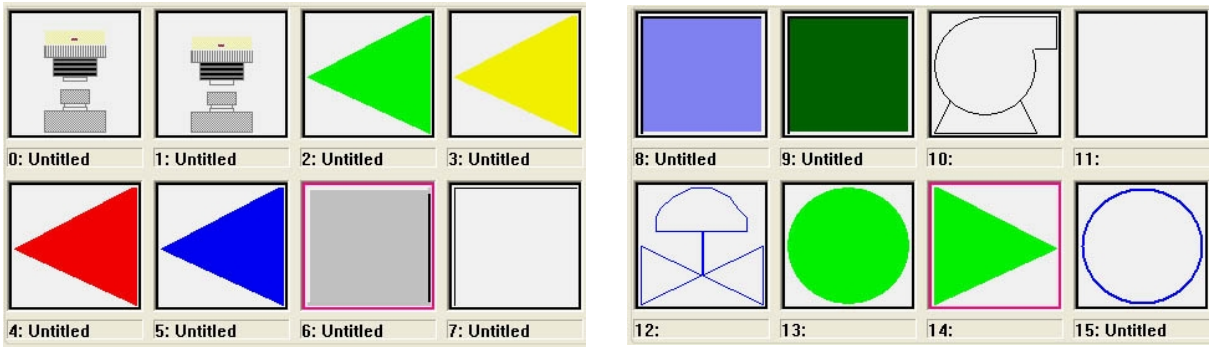
ISA_1.slb


0: Pump
1: Blower
2: Vessel1
3: Vessel2
4: Reactor
5: Bin
6: Hopper
7: Filter
40: Valve2
41: Valve3
42: Valve4
43: Valve5
44: Valve7
45: GateV
46: 4WaySol
47: 3SolRst
8: VacFilt
9: Air Ex
10: Furnace
11: RotKiln
16: RotFeed
17: Screw1
18: Mixer
19: Cmprssr
24: Untitled
25: CycSep
26: RotSep
27: Dryer
32: MixTee
33: Pitot
34: Venturi
35: Untitled
28: Blender
29: RollBnd
30: Statmix
31: Mixer
36: PDMeter
37: VortexM
38: TargetM
39: Valve1

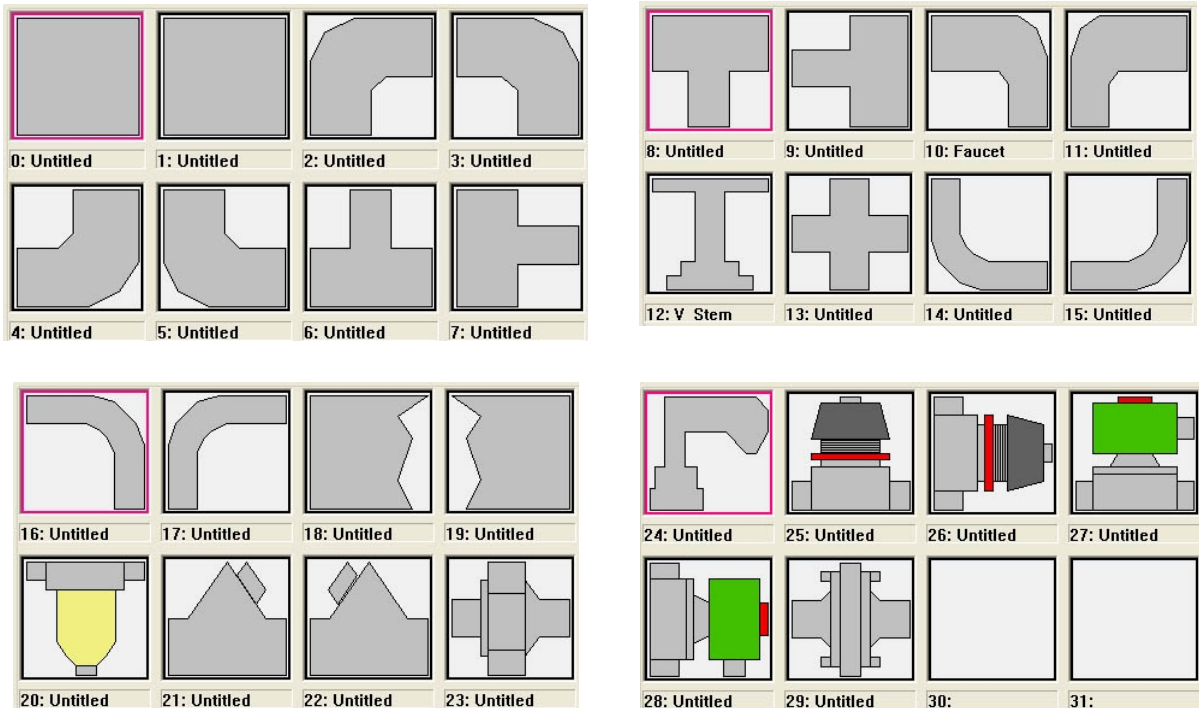
ISA_2.slb



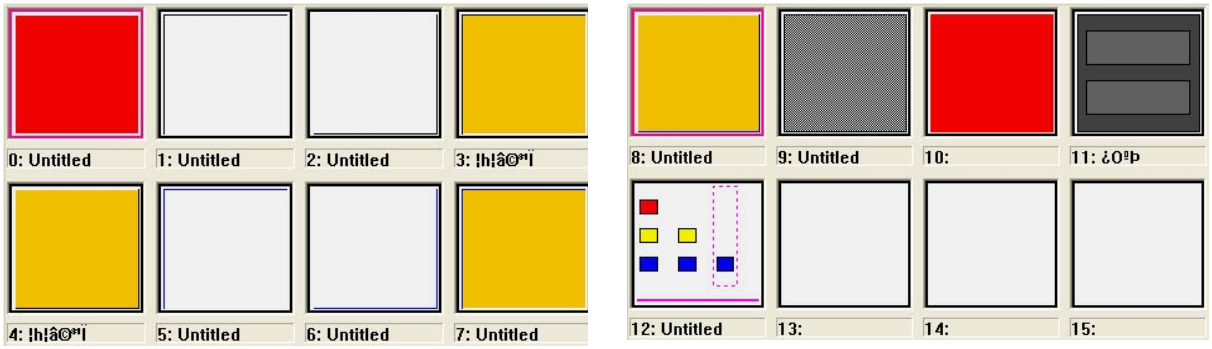
PART1.slb



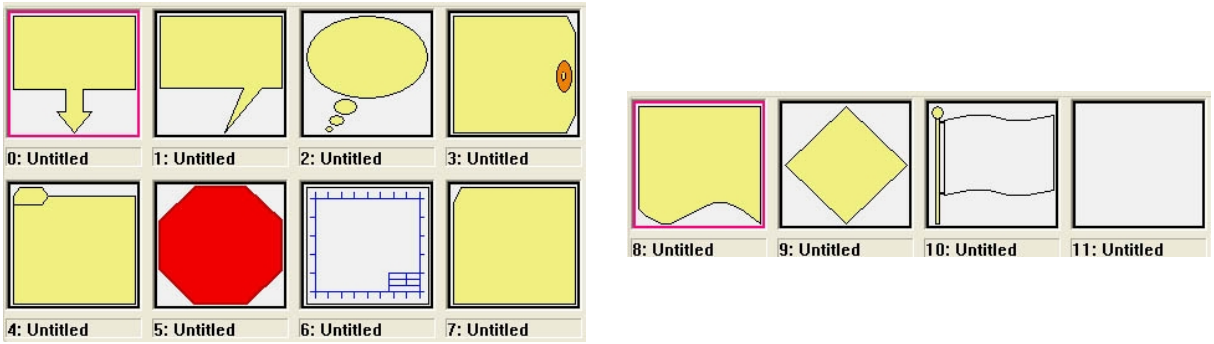
Pipes.slb


0: Untitled
1: Untitled
2: Untitled
3: Untitled
4: Untitled
5: Untitled
6: Untitled
7: Untitled
8: Untitled
9: Untitled
10: Faucet
11: Untitled
12: V Stem
13: Untitled
14: Untitled
15: Untitled
16: Untitled
17: Untitled
18: Untitled
19: Untitled
20: Untitled
21: Untitled
22: Untitled
23: Untitled
24: Untitled
25: Untitled
26: Untitled
27: Untitled
28: Untitled
29: Untitled
30:
31:

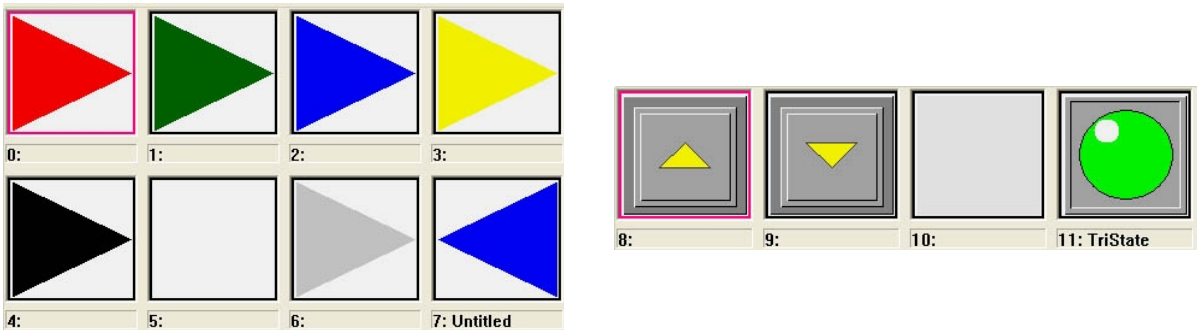
user-other.slb



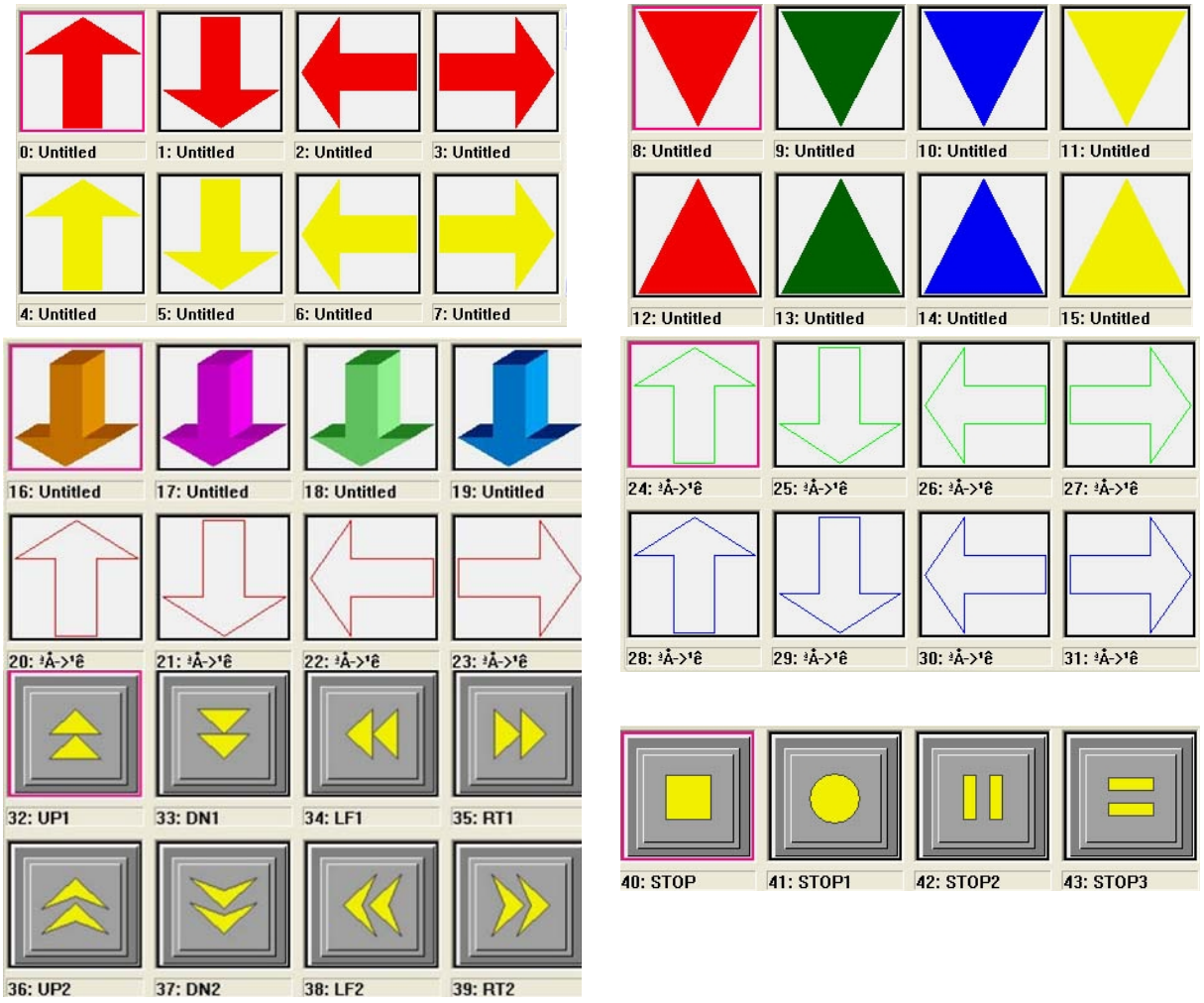
balloons.slb



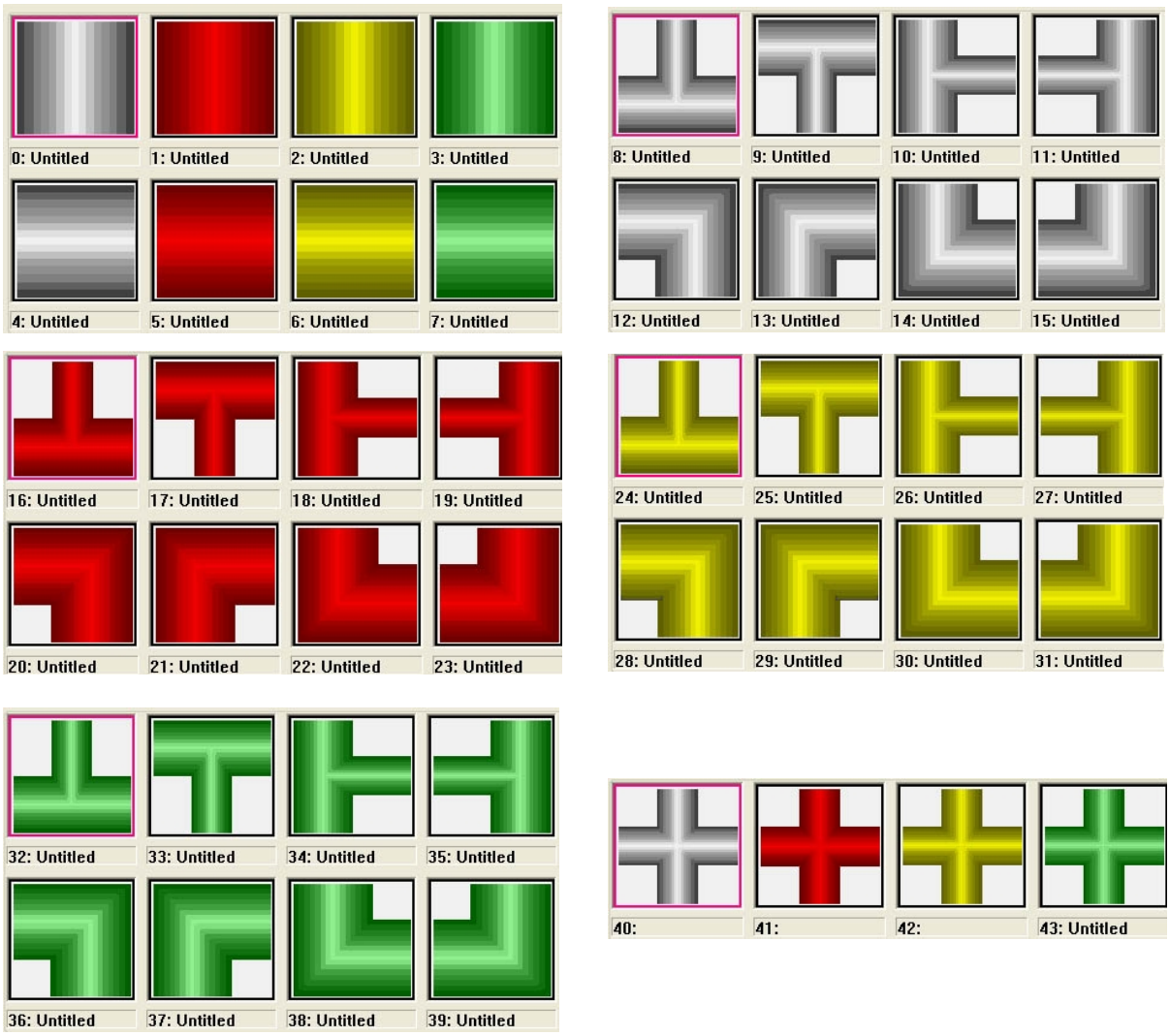
brewdemo.slb



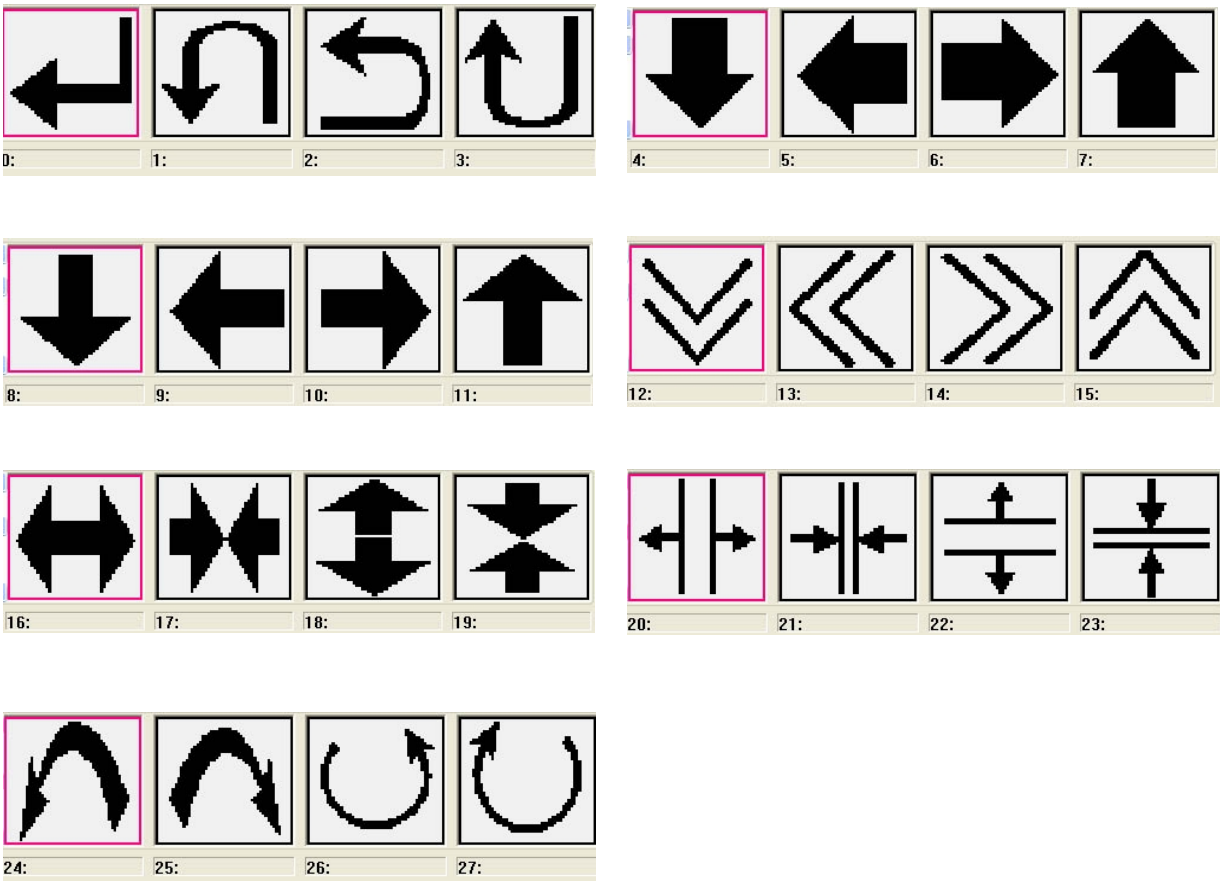
user-shape-arrow.slb



user-shape-pipe.slb



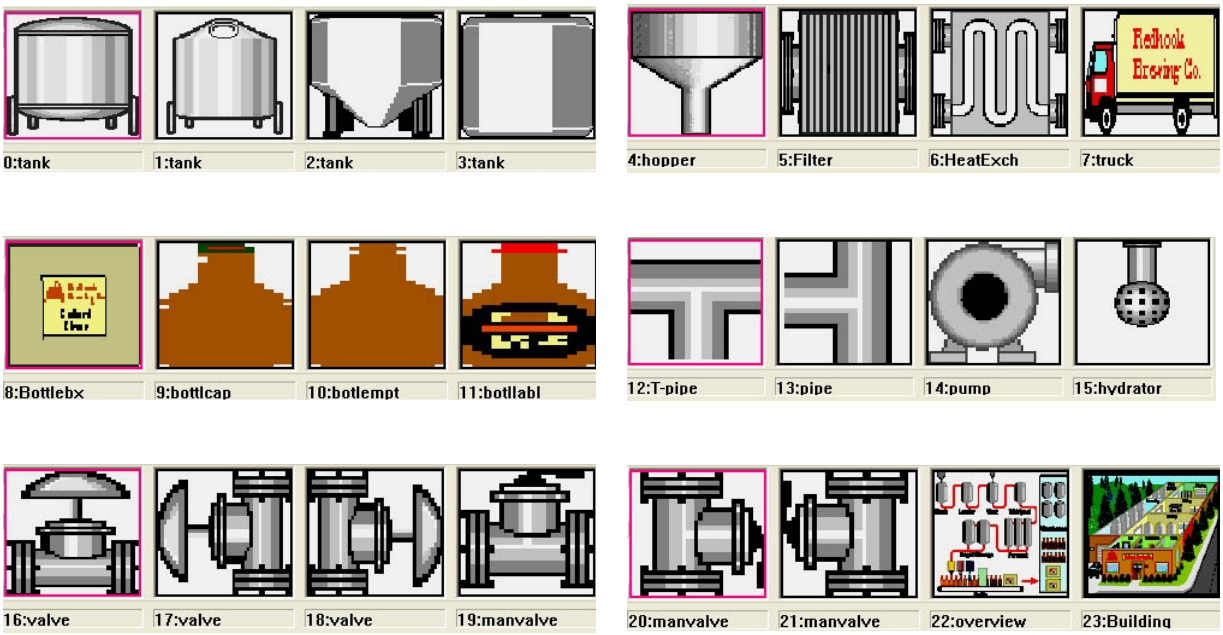
arrows.blb



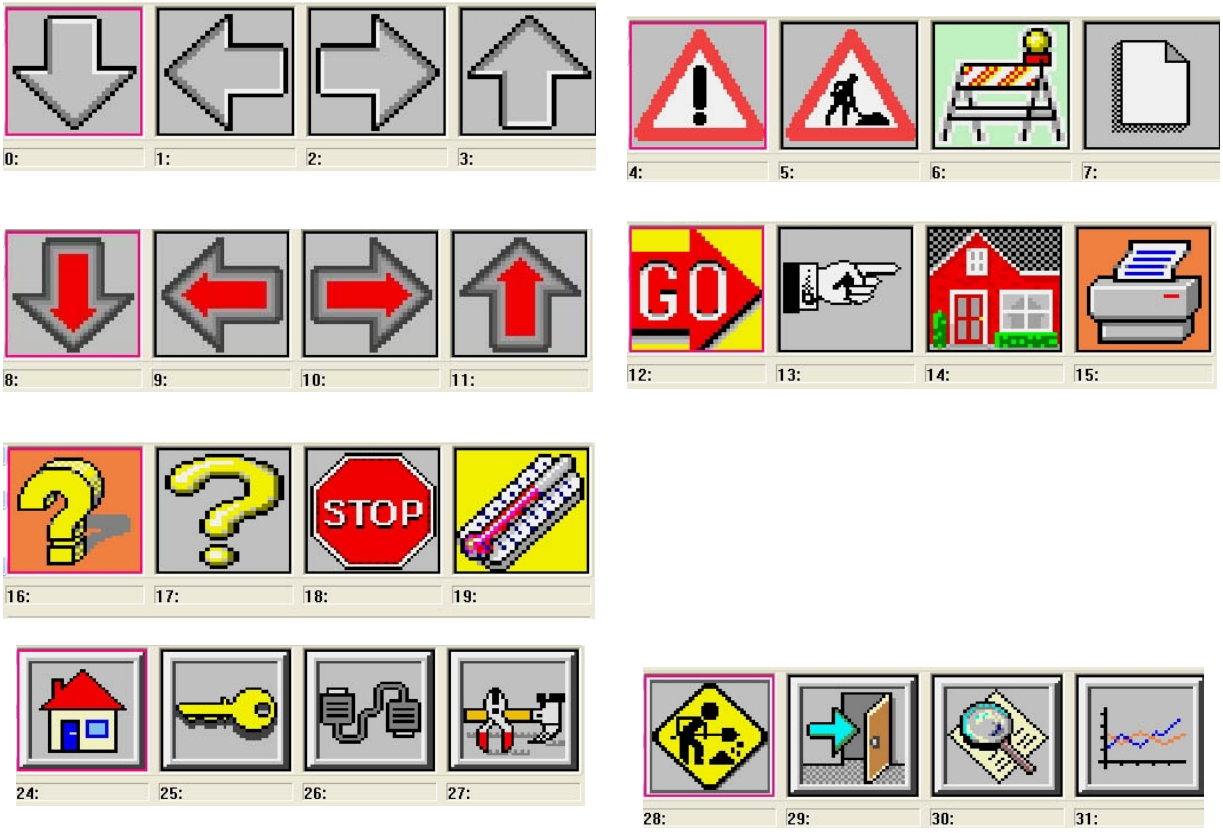
BMP1.blb

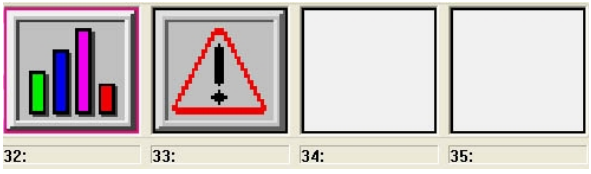

ON
OFF
ON
OFF
0:
1:
2:
3:
4:
5:
6:minbutt
7:
8:
9:
10:
11:
12:gage
13:
14:
15:
Redhook Brewing Co.
0:BotLbl6
1:BotFC6
2:Encl1
3:Encl2
4:Empty
5:Gas
6:Beer
7:Capped
8:Labled
9:BotFC6s
10:Bot5s
11:BotLbl6s
12:Bt6Empty
13:Bt6Gas
14:Bt6Beer
15:Bt6Cap
16:Bt6Lable
17:
18:Bt6p5s
19:

brewdemo.blb

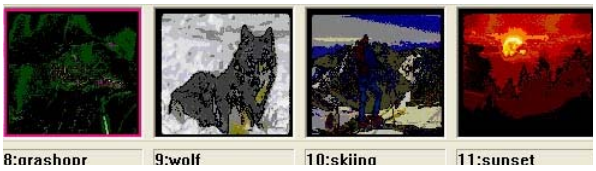


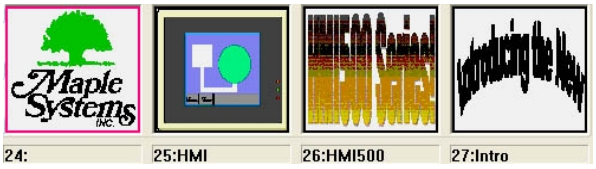
Icons.blb

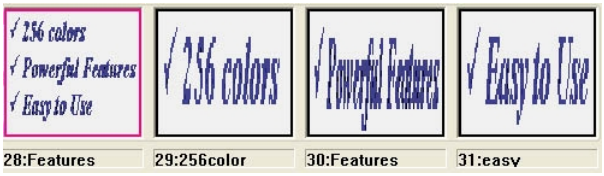




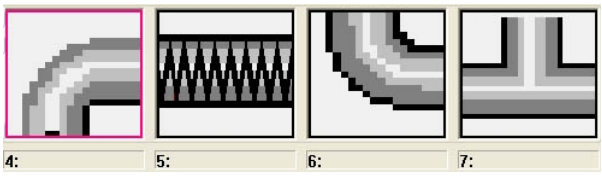
Pictures.blb

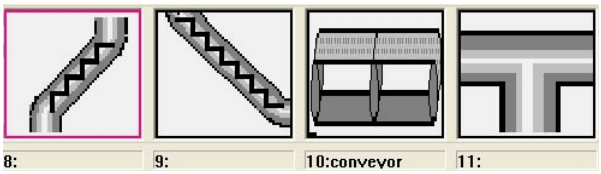


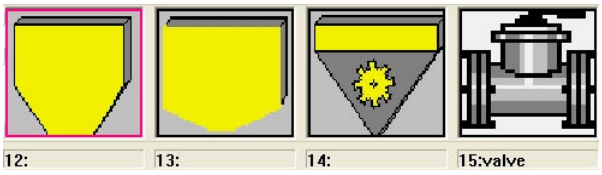


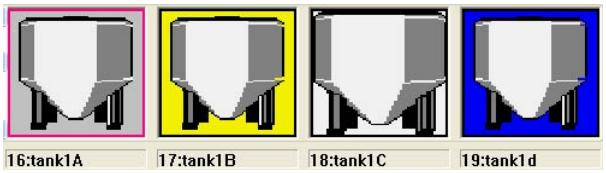


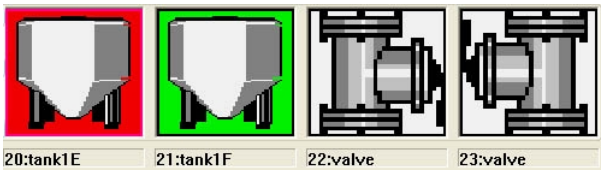
pipline.blb

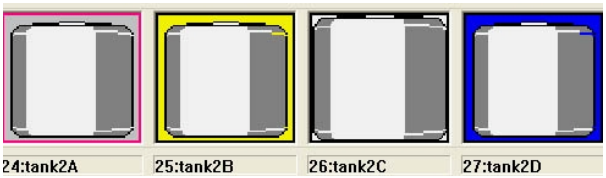


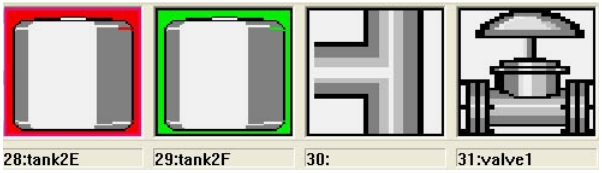

8:
9:
10:conveyor
11:


12:
13:
14:
15:valve

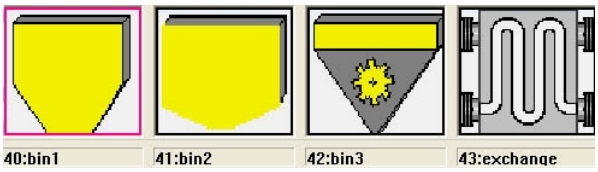

16:tank1A
17:tank1B
18:tank1C
19:tank1d

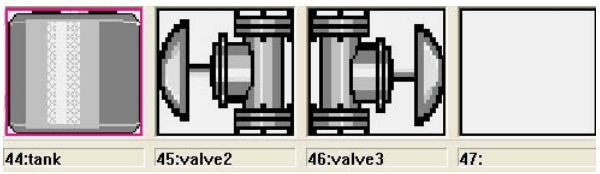

20:tank1E
21:tank1F
22:valve
23:valve


24:tank2A
25:tank2B
26:tank2C
27:tank2D


28:tank2E
29:tank2F
30:
31:valve1


36:tank3E
37:tank3F
38:pump
39:filter

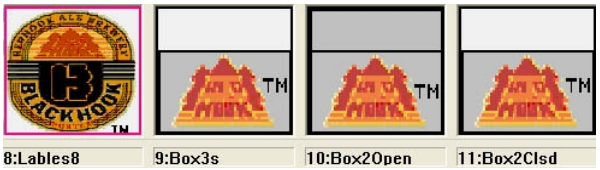

40:bin1
41:bin2
42:bin3
43:exchange


44:tank
45:valve2
46:valve3
47:

redhook.blb


0:Black
1:Blonde
2:DblBlk
3:ESB


4:Hefe
5:IPA
6:NutBrn
7:Winter

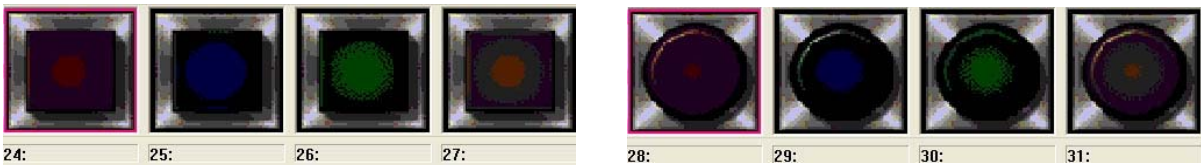

8:Lables8
9:Box3s
10:Box2Open
11:Box2Clsd

SWITCH1.blb


0:SSwitch
1:TSwitch
2:LSwitch
3:NFSwitch
HI
OFF
LO
4:3PSwitch
5:FSwitch
6:Switch
7:PSwitch

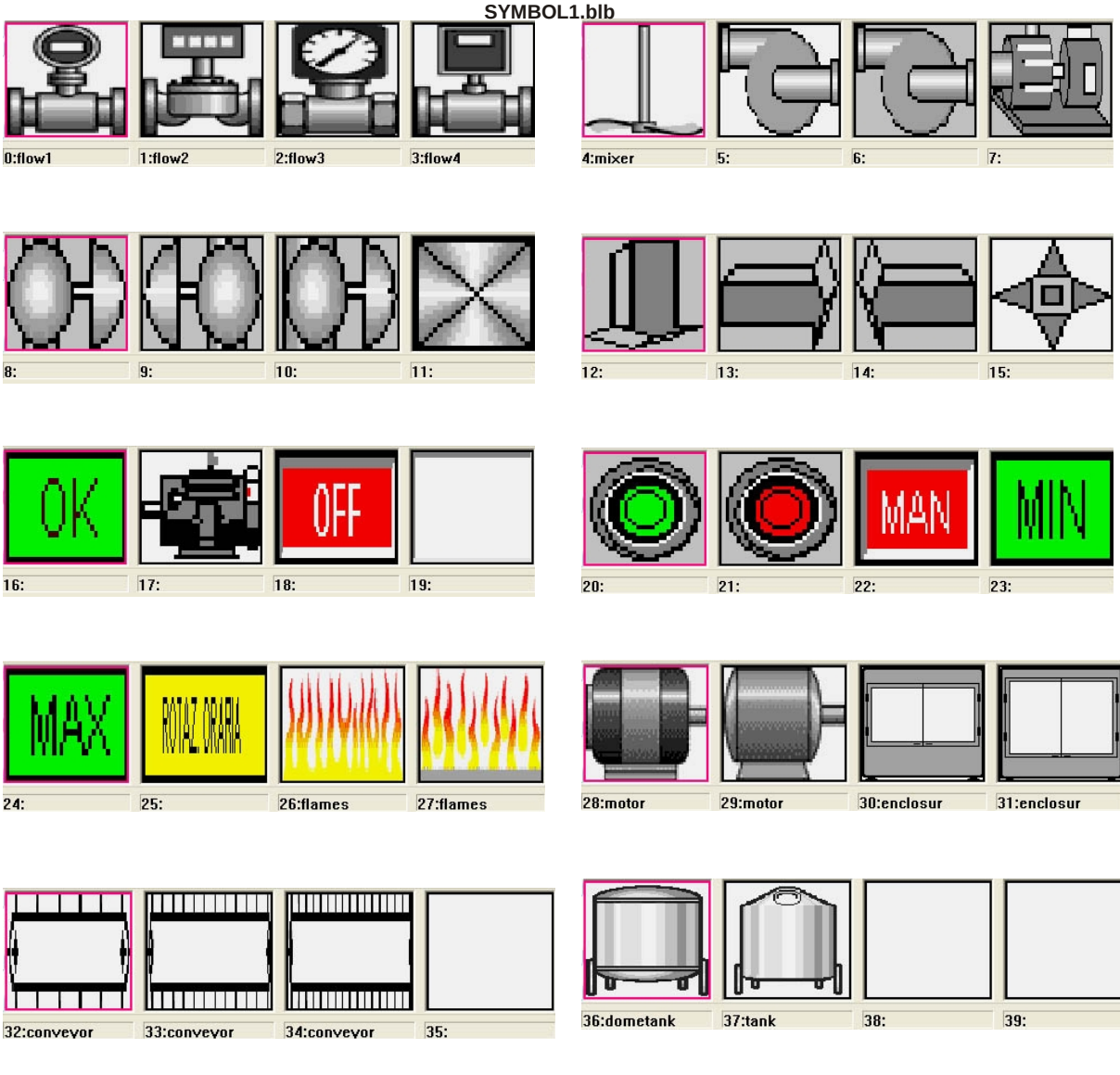

8:TLight
9:
10:1/2Meter
11:
12:
13:
14:
15:


16:
17:
18:
19:
20:
21:
22:
23:

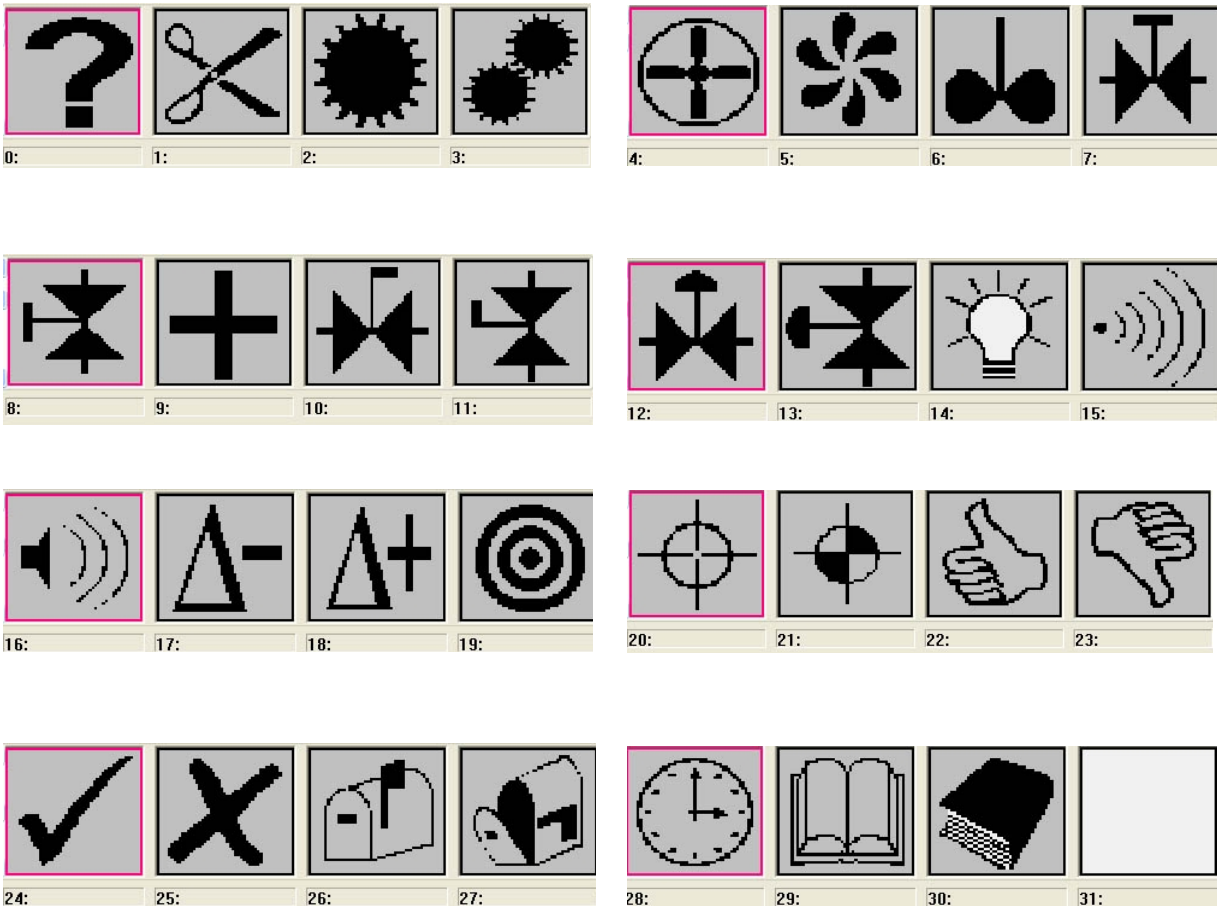

24:
25:
26:
27:
28:
29:
30:
31:


32:
33:
34:
35:
36:
37:
38:
39:
40:
41:
42:
43:
44:
45:
46:
47:

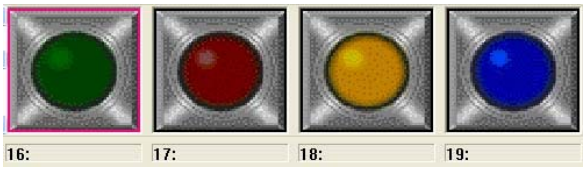
SYMBOL1.blb

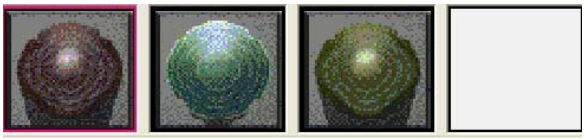

0:flow1
1:flow2
2:flow3
3:flow4
4:mixer
5:
6:
7:
8:
9:
10:
11:
12:
13:
14:
15:
OK
16:
17:
OFF
18:
19:
MAN
MIN
20:
21:
22:
23:
MAX
ROTAZ ORARIA
24:
25:
26:flames
27:flames
28:motor
29:motor
30:enclosur
31:enclosur
32:conveyor
33:conveyor
34:conveyor
35:
36:dometank
37:tank
38:
39:

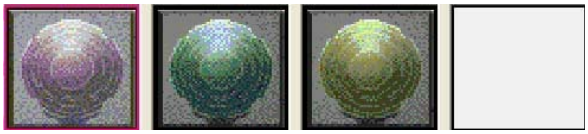
SYMBOL2.blb

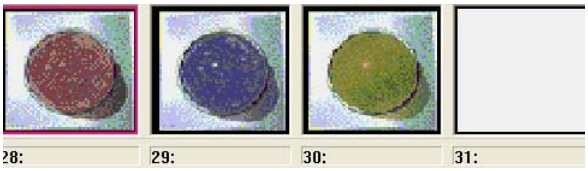


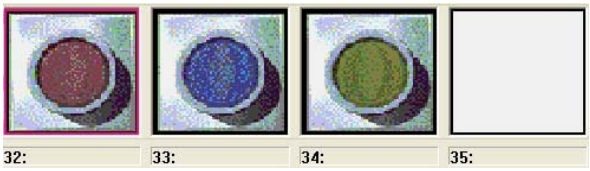
user-button.blb

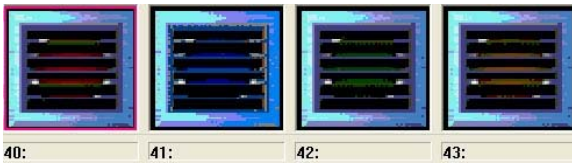




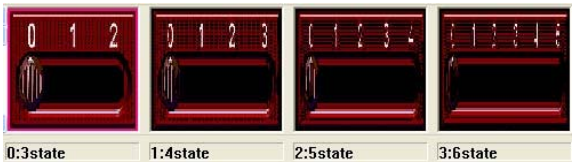


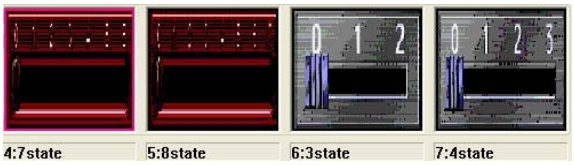


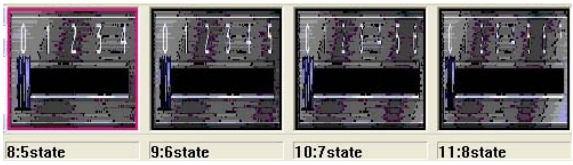




user-slide.blb







Index

A

Alarm Bar Object, using 268
Alarm Display Object, using 262
Alarm History, creating. 275
Alarm Indicator 75,141
Alarm Scan Object, using 261
Alarms, creating 261
Alarms, scrollable window 264
align bottom 63
align horizontal cente 63
align left 61
align right 62
align top 62
align vertical center 64
Alignment tools 61
Always ON bits. 188
Animation Object, using 226
Arc tool, using 148
ASCII Data Object, using 218
ASCII Input Object, using 221
assign Window Security Level passwords 107
Attributes, changing 57
Automatic toggling feature (periodical toggle). . 198

B

background pattern 110
Bar Graphs, creating 247
Base windows 110
Bit Lamp Object, using. 194
Bitmap Libraries, deleting 170
Bitmap Libraries, using. 165
Bitmap Object, using 154
Bitmaps, compressing 170
Bitmaps, definition of 151
Bitmaps, exporting 171
Blinking feature 195
Boolean expressions 299

C

Cable Routing and Noise Immunity 12
CE Compliance 9
change language 54
change state 54
Circle tool, using 148
Clipping Feature 103
Closing projects 49
Closing windows or screens 96
Coherence Feature 105
Com Port. 23
Common windows, above/below the base screen feature. 127
Common windows, changing. 128
Common windows, displaying popup windows from 124
Common windows, using. 122
Communications Settings 44
Compiling 33
compress feature 49
Connect the HMI to the PLC 15
Connecting HMI to Computer 23
Connecting Multiple HMIs to one PLC 82
Contrast level, adjusting 21
Control panel grounding 11
Control panel, requirements 10
Copy command 56
create a new project 47
Creating a new window 96
Creating XY Plots 256

D

Data Transfer Object, using 281
decrement key 32
Deleting a part or object 57
DIP switches 21
Direct Window Object 114

Display Meter Object, using 251
Display Options 51
Displaying your project 34
Drawing tools, using 147

E

EasyASCIIFontMaker utility 45
EasyBuilder Application 47
EasyManager 43
Editing Commands 54
Electrostatic Discharge 9
Ellipse/Circle tool, using 148
Event Display Object, using 272
Event Log Object, using 270
Events, definition 272
exit EasyBuilder 49
Exiting EasyBuilder 49
extract a compressed project file 49
EZware-500 configuration software, installation . 23
EZware-500 software 43

F

Factory default settings 21
Fast Selection key 133
Fast Selection windows, changing 133
Fast Selection windows, using 130
Fix Objects 54
Flipping tools, using 67
Floating Point Registers Within Macros 296
Font files, using 45
font size, changing 151
Full screen windows, displaying 111
Function Key Object, creating keypads 231
Function Key Object, displaying windows 111

G

General Settings 72
Graphics Object 3
grid color 53
Grid function 53
grid size 53
Group Libraries, definition 171
Group Libraries, deleting a group member 175
Grouping objects 58

H

Hard Copy 307
Hardware Settings Configuration 79

I

increment key 31
Indicator Settings Configuration 74
Indirect Window Object 114
Installing the OIT 13
Internal OIT Memory 187

J

Jog feature 206

K

Keypad, creating 231
Keypad, displaying and using 235

L

Labeling an object 197
languages 185
Languages 177
Layering tools 59
LED Indicators 5
Libraries, Graphics using 155
Line tools, using 147
Lock 69

M

Macros 287
Macros, Array initialization 292
Macros, Memory Usage 291
Macros, Reserved Words 292
Macros, using 287
Macros, using with recipes 295
Macros, Variable Declarations 291
Macros, Variable Initialization 292
Message board, changing operation mode 143
Message board, changing pen color 144

Message board, changing pen style. 144
Message board, clearing 144
Message Board, using 143
meter display. 30
Minimize windows feature 120
Modify a font file 46
Monopoly feature. 102
Moving a part or object 57
Moving Shape Object, scaling feature 227
Moving Shape Object, using 223
Moving windows feature 119
Multi-copy command, using 56
Multiple copies of object on screen, making . . . 56
MultiState Switch Object. 207

N

NEMA rating 9
Noise Immunity 12
Nudge tools, using 61
Numeric Data Object, using 209
Numeric Input Object, using 213
numeric register 30

O

Object 3
Object Attributes, viewing. 57
Object ID tags 53
Object ID tags, enable/disable 53
Objects, creating & editing 50
off-line simulation mode. 38
OIT (Operator Interface Terminal) 2
on-line simulation mode 38
Opening projects. 49
Opening windows or screens 95
Operating Modes. 44
operating temperature 9

P

Panel Preparation 17
Password Protection 73
PLC block pack feature 80
PLC Control Object 112
PLC, displaying windows 112
Polygon tool, using 149
Popup Window, creating. 28
Popup windows, displaying 111
Port wiring 13,17
Power supply requirements 12
Printer. 305
Printing projects 50
Programming the OIT, wiring 19
Project. 3
Project, creating 23
Projects, Editing 47
Projects, opening. 47
Pulsing a bit, using Set Bit Object 198

R

Rectangle tool, using 147
Redo command 55
Reserved local bits 188
Reserved local words. 190
Reset switch 21
Resizing a part or object 57
Rotation tools, using 68

S

Safety Precautions 10
Same Size tools 65
Sample Project 25
Saving 33
Saving Projects 49
scale lines 29
Scale lines tool, using 149
Security Passwords 107
select all objects 55
select next object. 55
Selecting graphic objects 54
Set Bit Object, using 198
Set Word Object, using. 205
Shape libraries, using. 157

Shape Object, using 152
Shapes, definition of 151
Sharing Data 83
Simulation Mode 37
Simulation Mode, wiring 40
Simulation Screen 37
Simulation, direct mode 40
Simulation, troubleshooting 39
Snap Option 53
Specifications 309
Starting EZware500 24
Startup Window, creating 26
States, definition of 156
System Parameters 26
system requirements 23

T

Tag Databases 177
tag databases, label library 183
Tag databases, label library 180
Tag Databases, tag library 177
Tag databases, using the tag library 179
Tag Library, Import and export 178
Task Bar, settings 142
Task Bar, using 140
Text Box Object, using 150
Text fonts 45
Toggle Switch Object, using 199
Touch Indicator 74
Touchscreen, calibration of 21
Tracking feature 99
Transparent feature, bitmaps 169
Trend Display Object, using 254,257

U

Undo command 55

W

Window Copy 56
Window frames, creating 109
Window Name 98
Window No. Treebar, using to navigate 51
Window styles 99
Window, definition of 3
Window, deleting 110
Window, fundamental settings 97
Window, selecting a new background color . . . 109
Windows, how to display 111
Windows, minimize feature 120
Windows, moving feature 119
wiring diagrams 19
Wiring diagrams 17
Wiring multiple OITs 83

X

XY Plots 256

Z

zoom in 54