Chapter 15 HUMAN MACHINE INTERFACE

Introduction

Communications between processor and HMI (human machine interface) is an important subject as well as constructing an operator interface. The chapter includes procedures for attaching computers as HMI devices to the CompactLogix processor from A-B and the Siemens 1200 processor. Graphic control packages used are A-B's RSView ME and Siemens' WinCC. Other packages exist and were not excluded based on their capabilities. The ones used are among the more common and popular ones used today.

The following HMI panels communicating to PLC processors will be discussed in this chapter followed by a lab that can show advantages and disadvantages of each.

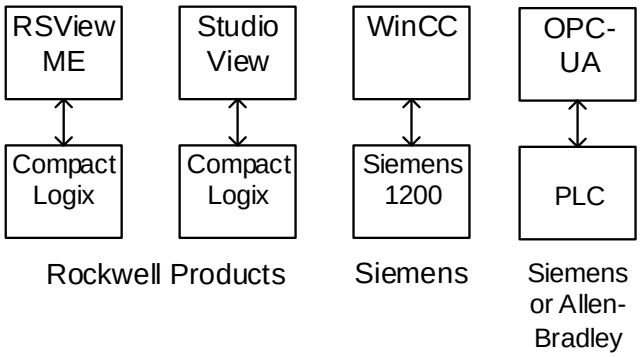


Fig. 15-1 HMI Graphic Control Packages

In each case, the emphasis is on getting a simple application operational and then expanding from that, remembering the analogy of the kite flying with a simple string over Niagara River. Later, the more difficult applications are discussed but only after a single button is programmed from the HMI and communicates successfully to the PLC.

Historical Panel Design

The design of an operator panel requires much coordination with the programming of the PLC and the design of the machine being controlled. Before the computer-designed systems, there were individual component systems that were hard-wired to the control devices inside the panel.

Fig. 15-2 A Simple Control Panel with Push Buttons and Switches with Indicator Lights

Fig. 15-3 This Printer is used for alarms for a process. Each alarm was recorded at the time of occurrence and printed as a single line of data to be analyzed by a process engineer or controls engineer.

Fig. 15-4 This panel shows many discrete devices as well as mimic panels showing process lines.

Meters show levels or flows of various devices.

Alarms are shown in grids of illuminated push buttons.

Fig. 15-5 Alarm panels were designed with discrete panels that lit or blinked with each alarm. Buttons were used to acknowledge each alarm point.

Fig. 15-6 Data was collected with recording devices similar to the above. Multiple points were individually recorded and studied.

Fig. 15-7 A handheld thermocouple readout device with paper recording output

Fig. 15-8 Several discrete controller devices for process control. Each device is capable of solving a single or multiple loops of process data executing a PID formula and controlling the output of the control loop.

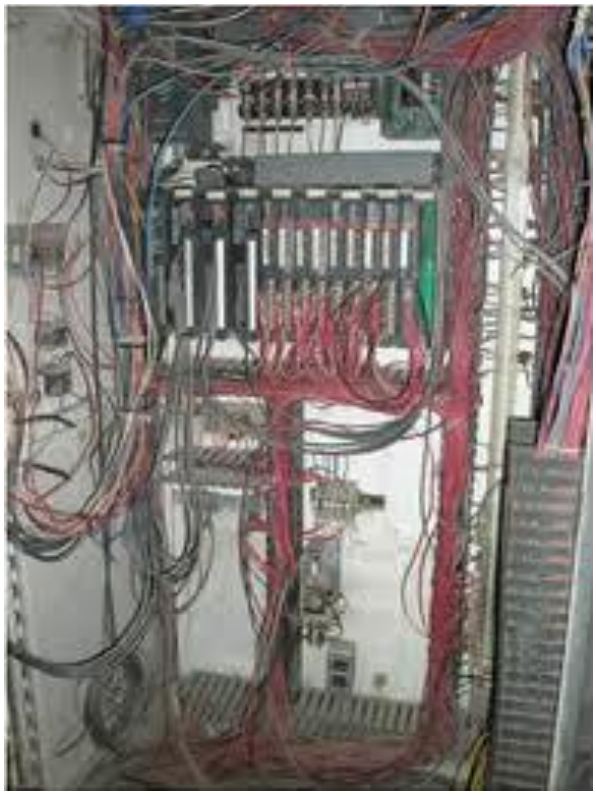
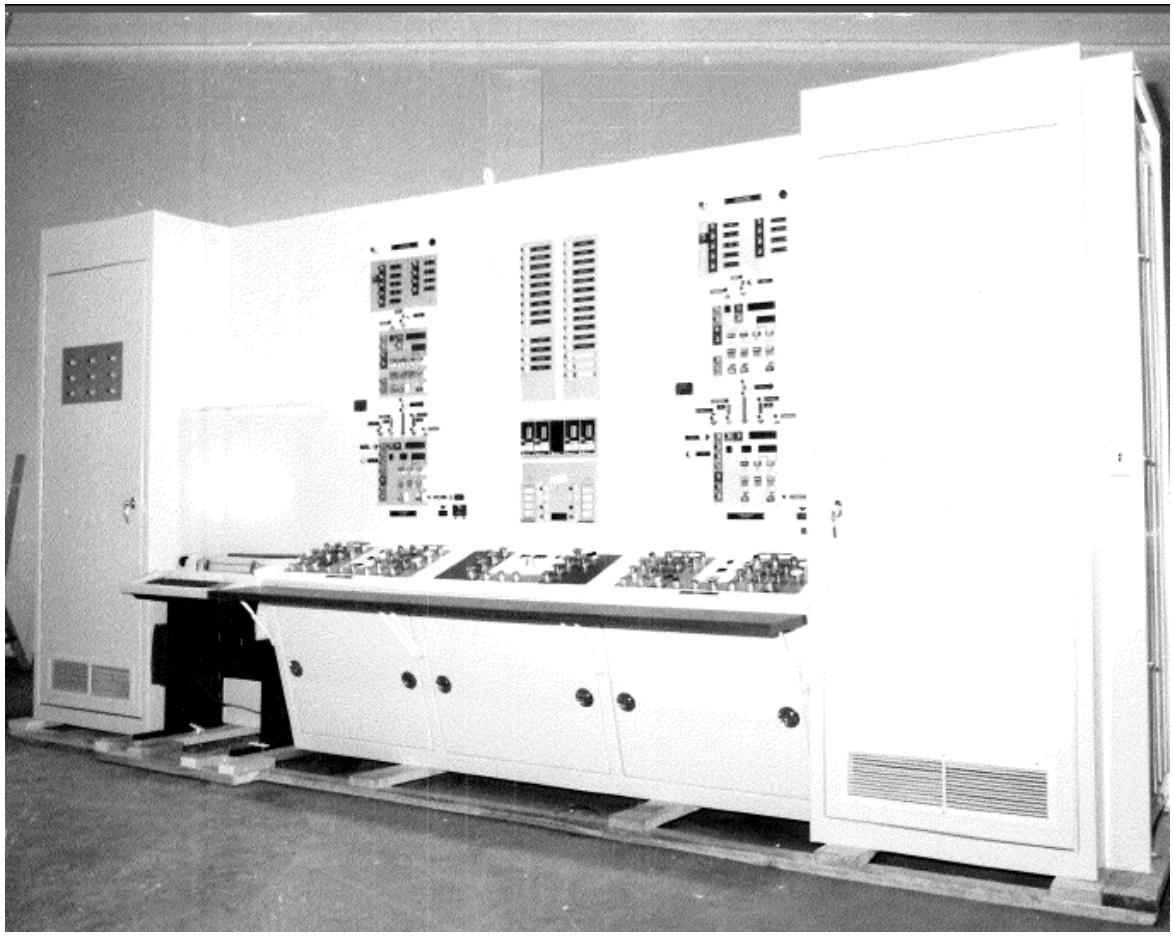


Fig. 15-9

Whatever the appearance of the outside of the panel, the inside many times looks similar to the panel shown at left. It is too easy for the panel to look like this after a short time even though the original plan showed a neat design with well-organized wiring layout. This is not just a rare bad example.

Fig. 15-10

The picture below is of a panel designed in 1980 for a chemical batching system. It should be seen as what 'was' and not as what today should be a good design. It cost approximately $100K then and was a divider between the operator room and the instrument control room for the process.



ISA-101 Human Machine Interfaces for Process Automation Systems

ISA-101 (officially ANSI/ISA-101.01-2015) is the HMI specification meant to describe proper development of an HMI panel. The specification is under development and should be referenced as it may have input for your HMI design.

Hardware to Complement the Software

The following Siemens panels are available for selection to replace the various panels above. Software allows the design of these 'soft' panels to be adapted to represent all the designs shown above. Siemens, which provides a complete offering of hardware operator interface units to complement the software requirements.

	KP8 PN	KP8F PN	KP32F PN
Type of operation			
Function keys (programmable)	8	8	32
Output type			
LED color modes	5 (green, red, yellow, blue, white)		
Typical service life			
Short-stroke keys (in number of switching cycles)	1,500,000		
Light-emitting diodes (ON period in %)	100%		
Interfaces			
Digital inputs / outputs[1)]	8	8	16
Fail-safe inputs SIL 2 / SIL 3	– / –	2 / 1	4 / 2
PROFINET	2	2	2
Functionality			
Pushbutton and lamp test	•		
Degree of protection			
Front / rear	IP65 / IP20		

Fig. 15-11 Siemens HMI

	KTP400 Basic	KTP700 Basic DP KTP700 Basic	KTP900 Basic	KTP1200 Basic DP KTP1200 Basic	KP300 Basic mono PN KP400 Basic color PN
Operating mode	**4" Touch + Key**	**7" Touch + Key**	**9" Touch + Key**	**12" Touch + Key**	**3.6" Key** **4" Key**
Display	Widescreen TFT, 65k colors, LED backlighting				FSTN-LCD Black&White Widescreen TFT
Size (in inches)	4.3"	7"	9"	12.1"	3.6" 4.3"
Resolution (W x H in pixels)	480 x 272	800 x 480	800 x 480	1,280 x 800	240 x 80 480 x 272
MTBF[6] backlighting (in h)	20,000	20,000	20,000	20,000	50,000
Front dimensions (in mm)	141 x 116	214 x 158	267 x 182	330 x 245	165 x 97 150 x 186
Operator controls	Touch screen and tactile keys	Touch screen and tactile keys	Touch screen and tactile keys	Touch screen and tactile keys	Tactile keys
Function keys (programmable) / system keys	4 / –	8 / –	8 / –	10 / –	10 / • 8 / •
Usable memory					
User memory	10 MB	10 MB	10 MB	10 MB	1 MB
Memory for options / recipes[4]	– / 256 KB	– / 256 KB	– / 256 KB	– / 256 KB	– / 40 KB
Alarm buffer	•	•	•	•	•
Interfaces					
Serial / MPI / PROFIBUS DP / PROFINET (Ethernet)	– / – / – / •	• [3] / • / • / – – / – / – / •	– / – / – / •	• [3] / • / • / – – / – / – / •	– / – / – / •
USB host / USB device	1 / –	1 / –	1 / –	1 / –	–
Slot for CF / Multimedia/SD	– / – / –	– / – / –	– / – / –	– / – / –	– / – / –
Functionality (when configured with WinCC TIA Portal)					
Signaling system (number of messages / message classes)	1,000 / 32	1,000 / 32	1,000 / 32	1,000 / 32	200 / 32
Process pictures	100	100	100	100	50
Tags	800	800	800	800	250 500
Vector graphics	•	•	•	•	•
Bar charts / trend diagrams	• / f(t)	• / f(t)	• / f(t)	• / f(t)	• / f(t)
Faceplates	–	–	–	–	–
Recipes	50	50	50	50	5
Archiving / Visual Basic scripts	• / –	• / –	• / –	• / –	– / –
PG functions	–	–	–	–	–

Fig. 15-12 Siemens HMI

SIMATIC HMI Comfort Panels						
KTP400 Comfort KP400 Comfort	TP700 Comfort KP700 Comfort	TP900 Comfort KP900 Comfort	TP1200 Comfort KP1200 Comfort	TP1500 Comfort KP1500 Comfort	TP1900 Comfort	TP2200 Comfort
4" Touch + Key **4" Key**	**7" Touch** **7" Key**	**9" Touch** **9" Key**	**12" Touch** **12" Key**	**15" Touch** **15" Key**	**19" Touch**	**22" Touch**
Widescreen TFT, 16 million colors, LED backlighting						
4.3"	7"	9"	12.1"	15.4"	18.5"	21.5"
480 x 272	800 x 480	800 x 480	1,280 x 800	1,280 x 800	1,366 x 768	1,920 x 1,080
80,000	80,000	80,000	80,000	80,000	50,000	30,000
140 x 116 152 x 188	214 x 158 308 x 204	274 x 190 362 x 230	330 x 241 454 x 289	415 x 310 483 x 310	483 x 337	560 x 380
Touch screen or tactile keys	Touch screen or tactile keys	Touch screen or tactile keys	Touch screen or tactile keys	Touch screen or tactile keys	Touch screen	Touch screen
4 (with LED) / – 8 (with LED) / •	– / – 24 (with LED) / •	– / – 26 (with LED) / •	– / – 34 (with LED) / •	– / – 36 (with LED) / •	– / –	– / –
4 MB	12 MB	12 MB	12 MB	24 MB	24 MB	24 MB
4 MB / 512 KB	12 MB / 2 MB	12 MB / 2 MB	12 MB / 2 MB	24 MB / 4 MB	24 MB / 4 MB	24 MB / 4 MB
•	•	•	•	•	•	•

Fig. 15-13 Siemens HMI

• [3] / • / • / 1	• [3] / • / • / 2	• [3] / • / • / 2	• [3] / • / • / 2	• [3] / • / • / 3	• [3] / • / • / 3	• [3] / • / • / 3
1 / 1	2 / 1	2 / 1	2 / 1	2 / 1	2 / 1	2 / 1
– / • / •	– / • / •	– / • / •	– / • / •	– / • / •	– / • / •	– / • / •
2,000 / 32	4,000 / 32	4,000 / 32	4,000 / 32	6,000 / 32	6,000 / 32	6,000 / 32
500	500	500	500	750	750	750
1,024	2,048	2,048	2,048	4,096	4,096	4,096
•	•	•	•	•	•	•
• / f(t), f(x)	• / f(t), f(x)	• / f(t), f(x)	• / f(t), f(x)	• / f(t), f(x)	• / f(t), f(x)	• / f(t), f(x)
•	•	•	•	•	•	•
100	300	300	300	500	500	500
• / •	• / •	• / •	• / •	• / •	• / •	• / •
STATUS / CONTROL, diagnostics viewer						

SIMATIC HMI Mobile Panels				
2nd Generation	2nd Generation	2nd Generation		
KTP400F Mobile	KTP700 Mobile KTP700F Mobile	KTP900 Mobile KTP900F Mobile	Mobile Panel 277 IWLAN V2 Mobile Panel 277 (F) IWLAN V2	
4" Touch + Key	7" Touch + Key	9" Touch + Key	8" Touch + Key	**Operating mode**
Widescreen TFT, 16 million colors, LED backlighting			TFT display (LCD), 64k colors	**Display**
4.3"	7"	9"	7.5"	Size (in inches)
480 x 272	800 x 480	800 x 480	640 x 480	Resolution (W x H in pixels)
50,000	50,000	50,000	50,000	MTBF [6] back- lighting (in h)
194 x 166	248 x 172 248 x 195	307 x 201 307 x 224	Diameter 290	Front dimensions (in mm)
Touch screen and tactile keys	Touch screen and tactile keys	Touch screen and tactile keys	Touch screen and tactile keys	**Operator controls**
4 (with LED) / –	8 (with LED) / –	10 (with LED) / –	18 / –	Function keys (program- mable) / system keys
				Usable memory
4 MB	12 MB	12 MB	6 MB	User memory
4 MB / 512 KB	12 MB / 2 MB	12 MB / 2 MB	1,024 KB / 64 KB	Memory for options / recipes [4]
•	•	•	•	Alarm buffer
				Interfaces
– / – / – / 1	– / – / – / 1	– / – / – / 1	– / – / – / (• via IWLAN)	Serial / MPI / PROFIBUS DP / PROFINET (Ethernet)
1 / –	1 / –	1 / –	• / –	USB host / USB device
– / • / •	– / • / •	– / • / •	– / • / •	Slot for CF / Multimedia / SD
				Functionality (when configured with WinCC TIA Portal)
2,000 / 32	4,000 / 32	4,000 / 32	4,000 / 32	Signaling system (number of messages / message classes)
500	500	500	500	Process pictures
1,024	2,048	2,048	2,048	Tags
•	•	•	•	Vector graphics
• / f(t), f(x)	• / f(t), f(x)	• / f(t), f(x)	• / • f(t)	Bar charts / trend diagrams
•	•	•	•	Faceplates
100	300	300	300	Recipes
• / •	• / •	• / •	• / •	Archiving / Visual Basic scripts
STATUS / CONTROL, diagnostics viewer			STATUS / CONTROL	PG functions

Fig. 15-14 Siemens HMI

General features	Panel PC, 7" Touch	Panel PC, 9" Touch	Panel PC, 12" Touch	Panel PC, 15" Touch	Panel PC, 19" Touch
Resolution in pixels (widescreen)	(800 x 480)	(800 x 480)	(1,280 x 800)	(1,280 x 800)	(1,366 x 768)
Processor	Intel Celeron N2807 (2C/2T, 1.58 (2.16) GHz, 1 MB cache, VT-x); Intel Celeron N2930 (4C/4T, 1.83 (2.16) GHz, 2 MB cache, VT-x)				
Main memory	2 GB, 4 GB or 8 GB; 512 KB NVRAM optional				
Free expansion slots	–				
Operating systems (preinstalled and activated)	Windows Embedded Standard 7 (E/P), 32-bit/64-bit; Windows 7 Ultimate, MUI[1], 32-bit/64-bit				
Packages / bundles	Packages with WinCC RT Advanced, WinCC V7 and WinAC RTX (F)				
Power supply / temporary voltage interruption	24 V DC; 20.4 ... 28.8 V; isolated / max. 10 ms (according to NAMUR); On/Off switch				
MTBF backlighting	up to 80,000 h[2]; dimmable from 0 to 100%				up to 50,000 h[2]
Drives					
Mass storage	CFast up to 16 GB (accessible from outside); SSD 80 / 160 GB; HDD 320 GB (IPC227E only)				
Optical drives	–				
Interfaces					
Fieldbus	PROFINET RT over Ethernet				
Ethernet	2 x 10/100/1000 Mbps (RJ45); teaming				
USB	Rear: 1 x USB 3.0, 2 x USB 2.0		Rear: 1 x USB 3.0, 3 x USB 2.0	Rear: 1 x USB 3.0, 3 x USB 2.0; front: USB 2.0	
Serial / parallel	1 x RS 232/RS 485/RS 422 can be selected in BIOS				
Graphics interface	1 x DisplayPort				

SIMATIC IPC477D

Panel PC, 12" Touch	Panel PC, 15" Touch or Multitouch	Panel PC, 15" Touch/Key	Panel PC, 19" Touch or Multitouch	Panel PC, 22" Touch or Multitouch
(1,280 x 800)	(1,280 x 800)	(1,280 x 800)	(1,366 x 768)	(1,920 x 1,080)
Intel Core i7-3517UE (2C/4T, 1.7 (2.8) GHz, 4 MB cache, VT-xid, AMT); Intel Core i3-3217UE (2C/4T, 1.6 GHz, 3 MB cache, VT-x); Intel Celeron 827E (1.4 GHz, 1.5 MB cache, VT-x)				
1 GB, 2 GB, 4 GB or 8 GB; 512 KB NVRAM optional				
–	1 x PCIe x 4 (optional); max. 5 W			
VxWorks Support Package, suited for Linux				
Bundles with WinCC RT Advanced and WinAC RTX (F); packages: WinCC RT Professional, WinCC V7, WinAC RTX (F)				
24 V DC; 19.2 ... 28.8 V; isolated / max. 20 ms (according to NAMUR); On/Off switch	24 V DC; 19.2 ... 28.8 V; isolated / max. 20 ms (according to NAMUR); or 100–240 V AC, 50/60 Hz; On/Off switch			
up to 80,000 h[2]; dimmable from 0 to 100%			up to 50,000 h	up to 30,000 h
CFast up to 16 GB (accessible from outside); SSD 80 / 160 GB	CFast up to 16 GB (accessible from outside); SSD 80 / 160 GB; HDD 320 GB			
–	DVD-RW (optional)			
PROFINET RT over Ethernet; PROFIBUS DP/MPI: 1 x 12 Mbps (isolated) optional; PROFINET IRT: 1 x 10/100 Mbps (3-port-switch, optional)				
2 x 10/100/1000 Mbps (RJ45); teaming; 1 x 10/100/1000 Mbps with PROFINET IRT variant				
Rear: 4 x USB 3.0	Rear: 4 x USB 3.0; front: 1 x USB 2.0 (with Touch version only)			
COM1: RS 232; COM2: RS 232 (optional)				
1 x DVI-I; 1 x DisplayPort				

SIMATIC IPC677D

Panel PC, 15", 19" or 22" Touch or Multitouch	General features
15" Touch: 1,366 x 768; 15" Multitouch: 1,280 x 800; 19": 1,366 x 768; 22": 1,920 x 1,080	Resolution in pixels (widescreen)
Intel Xeon E3-1268L v3 (4C/8T; 2.3 (3.3) GHz; 8 MB cache; VT-d; AMT 9.0); Core i3-4330TE (2C/4T; 2.4 GHz; 4 MB cache; VT-x); Celeron G1820TE (2C/2T; 2.2 GHz; 2 MB cache)	Processor
From 2 GB DDR3-1600 SDRAM; 2 x DIMM; configurable up to 16 GB; ECC optional; non-volatile memory: NVRAM 2 MB optional	Main memory
2 x PCI (240 mm) or 1 x PCIe x 16 (185 mm), 1 x PCI (185 mm) or 1 x PCIe x 16 (185 mm), 1 x PCIe x 4 (185 mm)	Free expansion slots
Windows 7 Ultimate (32/64-bit) MUI [1]; Windows Embedded Standard 7 P (32-bit); released for S7-1500 Software Controller, suited for Linux	Operating systems (preinstalled and activated)
Packages with WinCC V7; WinCC RT Advanced; WinCC RT Professional and WinAC RTX (F)	Packages / bundles
AC: 100–240 V; 50 – 60 Hz / max. 20 ms (according to NAMUR); 24 V DC: 20.4 ... 28.8 V	Power supply / temporary voltage interruption
up to 50,000 h	MTBF backlighting

Fig. 15-15 Siemens HMI

SIMATIC Industrial Monitors and Thin Clients			
	SIMATIC Industrial Thin Client	SIMATIC Industrial Flat Panel	
General features	**12", 15", 19" or 22" Touch**	**15" Touch or Multitouch, 15" Key**	**19" and 22" Touch or Multitouch**
Resolution in pixels (widescreen)	12" (1,280 x 800) 15" (1,280 x 800) 19" (1,366 x 768) 22" (1,920 x 1,080)	15" Multitouch: (1,366 x 768) 15" Touch: (1,280 x 800) 15" Key: (1,280 x 800)	19": (1,366 x 768) 22": (1,920 x 1,080)
Max. distance to PC	Unlimited over Ethernet	Standard: 5 m; extended: 30 m	Standard: 5 m; extended: 30 m
Processor	Intel Celeron (1.2 GHz)	–	
Operating system (preinstalled and activated) / supported protocols	Closed Linux / VNC; SINUMERIK; WinCC-OA; web browser; JAVA; CITRIX Client	–	–
Power supply / max. power consumption	12" approx. 28 W 15" approx. 36 W 19" approx. 32 W 22" approx. 53 W	15": 24 V DC; 19.2 ... 28.8 V, approx. 40 W	24 V DC; 19.2 ... 28.8 V, approx. 40 W
MTBF background lighting	up to 50,000 h[7]; dimmable from 0 to 100%	up to 50,000 h[7]; dimmable from 0 to 100%	up to 50,000 h[7]; dimmable from 0 to 100%

Fig. 15-16 Siemens HMI

The hardware above shows the variable nature of hardware panels available for use. The cell phone is also available with an app installed that communicates with the system software. Each device gives a level of access to the PLC program and give visual feedback to the engineer or plant personnel.

The Software for Development

We begin the software development of the three programs. Two programs exist in the A-B platform and one is available from Siemens. We will also discuss an OPC-UA application for a total of four different programming platforms.

Common Tags

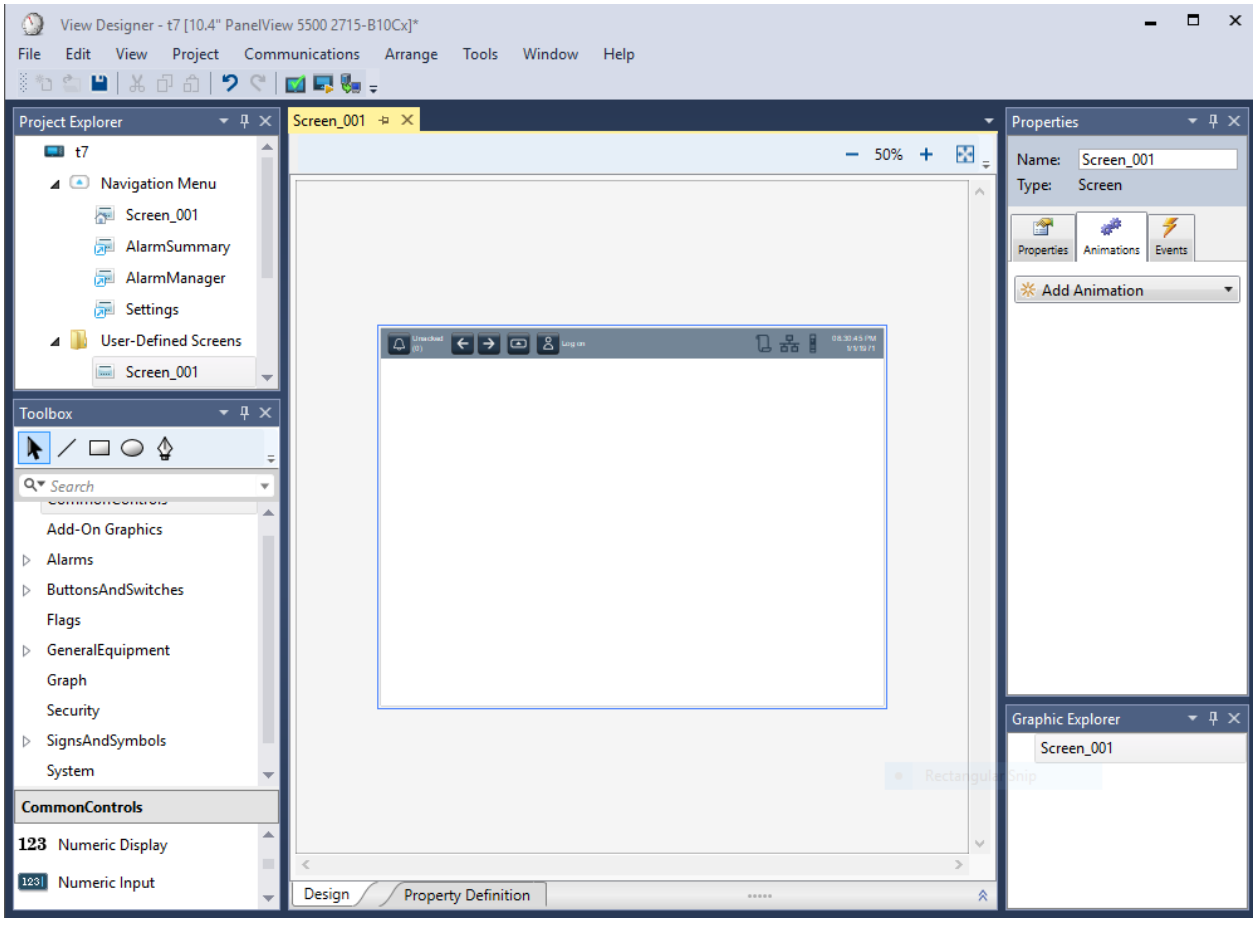
The most obvious advantage of using any new PLC platform is the universal accessibility of data tags. Tags created in any tool for any device are automatically and immediately accessible to other devices. If, for example, a user creates a new tag in the PLC to measure temperature, that tag is automatically created in the operator panel at the same time. This saves valuable engineering time compared to earlier methods that require the tag to be created in each device.

The suspension bridge at Niagara Falls was started by flying a kite with a string attached across the Niagara River. When wind conditions were favorable, the kite was flown across the river. Then a string was attached to the thread and a bridge was the eventual result. Likewise, programs in this chapter can be started with small threads and then expanded. It is best to get a simple device such as a button programmed and fully working and then adding the rest of the project after the button has been proved to thoroughly work in all modes

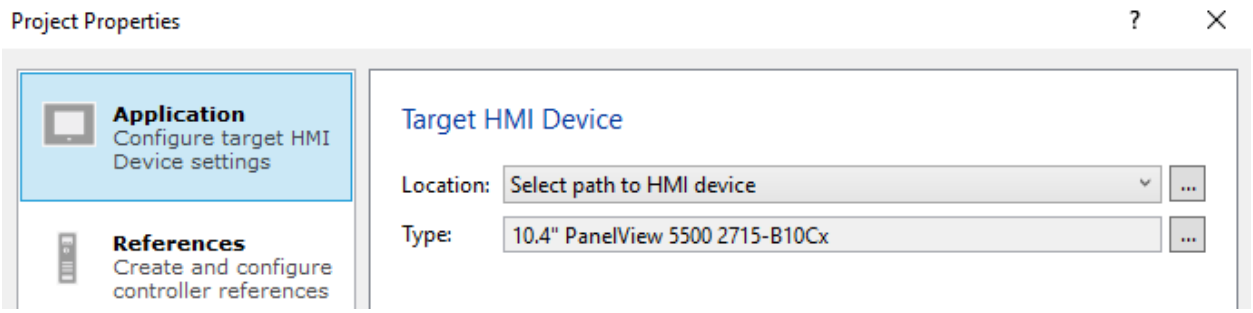
Allen-Bradley's New View Designer

Allen-Bradley recently released a new graphic HMI Development Environment. It is found on Studio 5000 as 'View'.

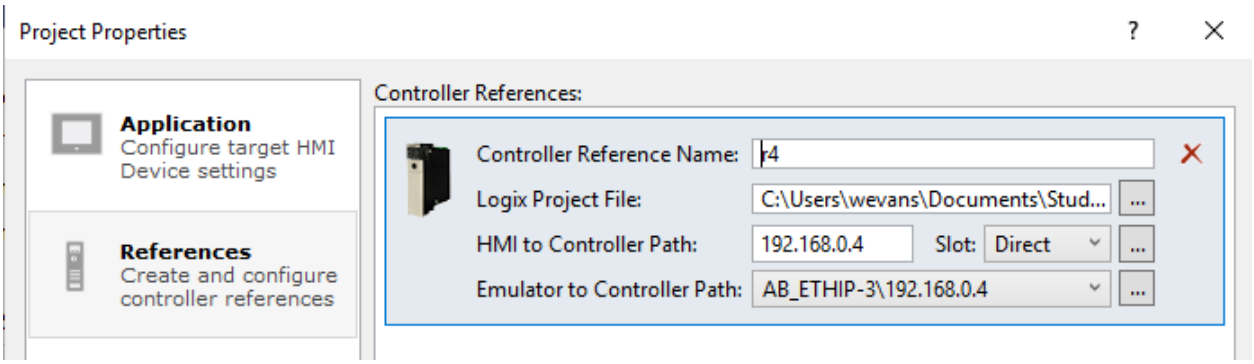
We began the View Designer with a project named 't7' tied to a plc program in a processor named 'r4'.



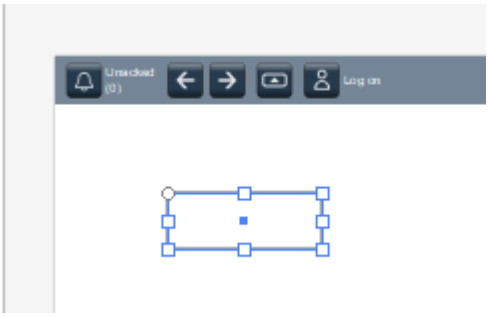
The plc application is already running and the application is selected under Project properties. We select a 10.4 “ PanelView 5500 panel although we are never going to download to this HMI panel. We plan to only use it in the emulate mode.



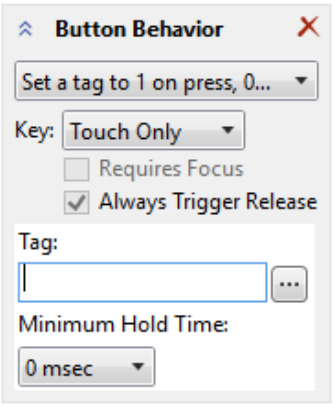
We then select the reference tab and tie the HMI to the processor ‘r4’ at IP address 192.168.0.4.



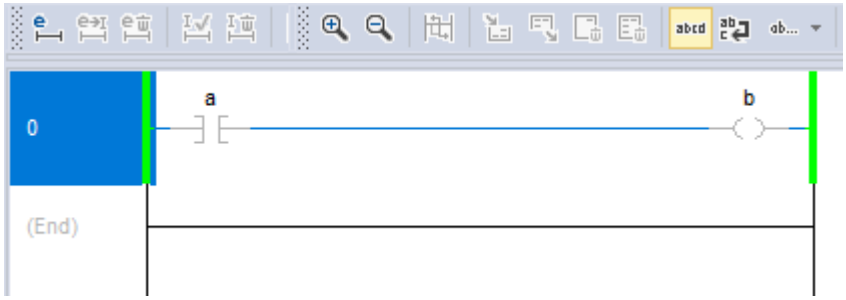
We then start with a button but use a rectangle instead of the pre-designed buttons.



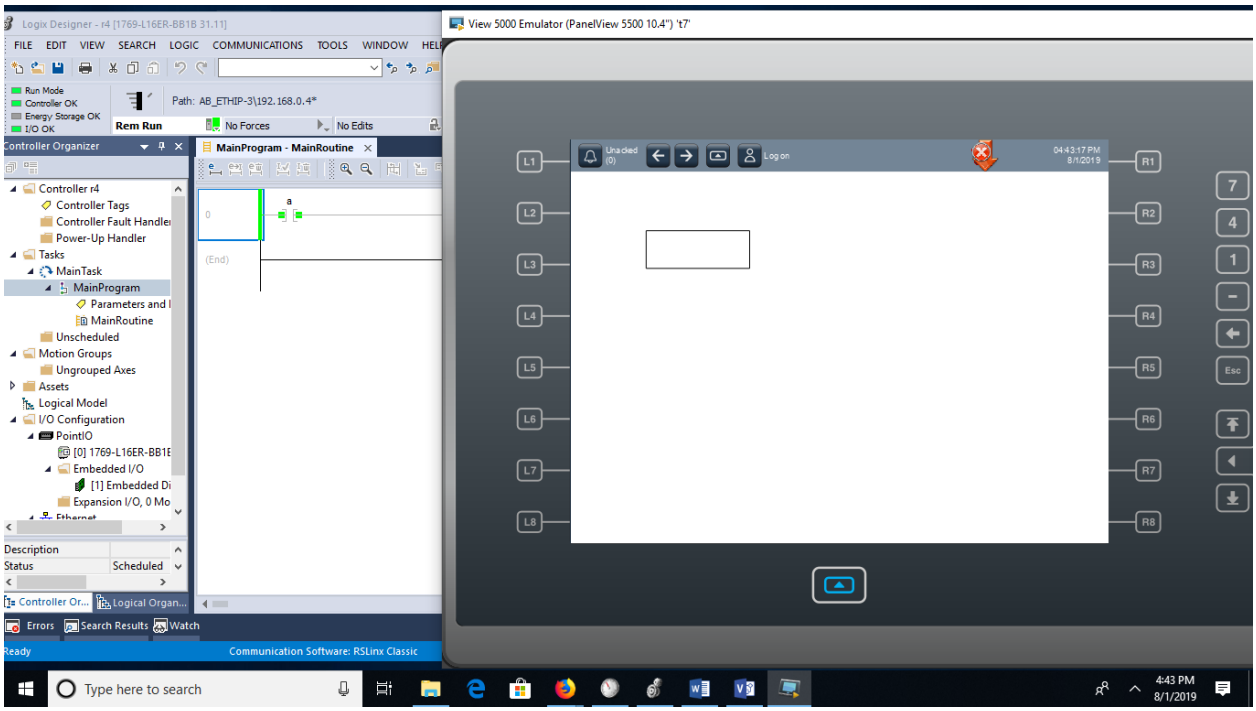
Right click on rectangle. Choose Set tag to 1 on press, 0 on release. Then below is a box for the tag. If attached properly above to the processor, the available bit may be seen in the Tag window. Choose the appropriate variable.



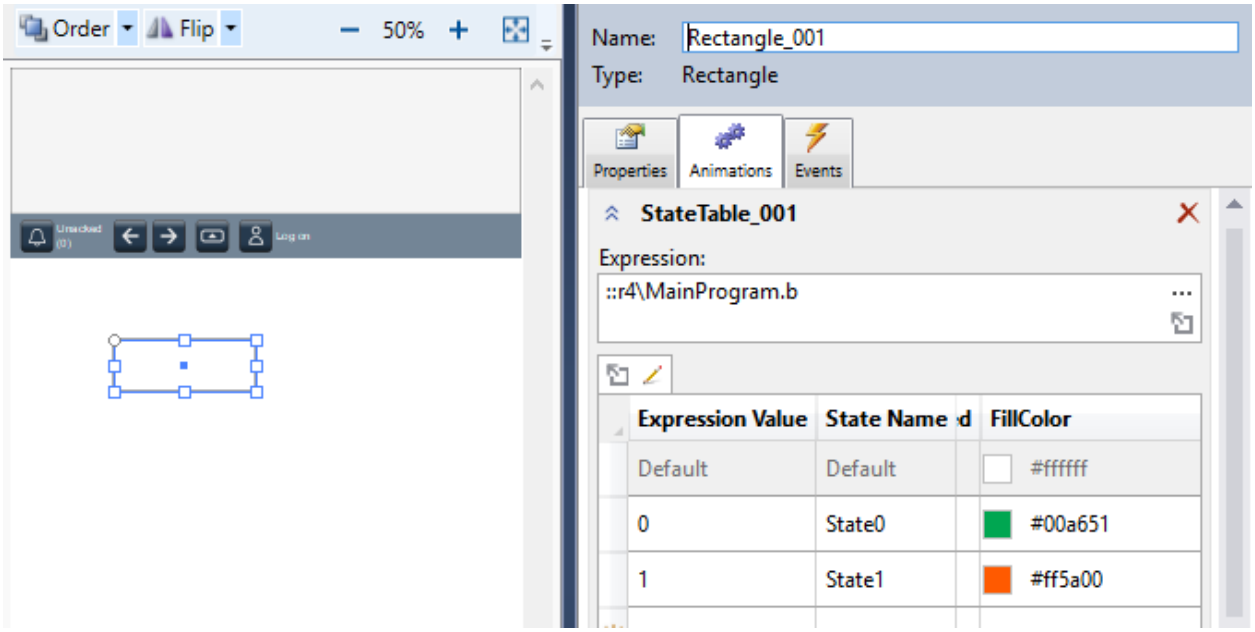
The tag we tied to resides in the PLC and is ready to be tested. If we were to toggle the button (rectangle) on and off, we should see the contact respond accordingly with the emulator running.



Under View 5000's Project tab, run the emulator program. With the emulator running, by pushing the rectangle with the mouse button, the bit 'a' should turn on. It does. This is a good test that the program is communicating properly with the HMI and gives you a positive feedback that you are ready to proceed with other logic development. The button works!



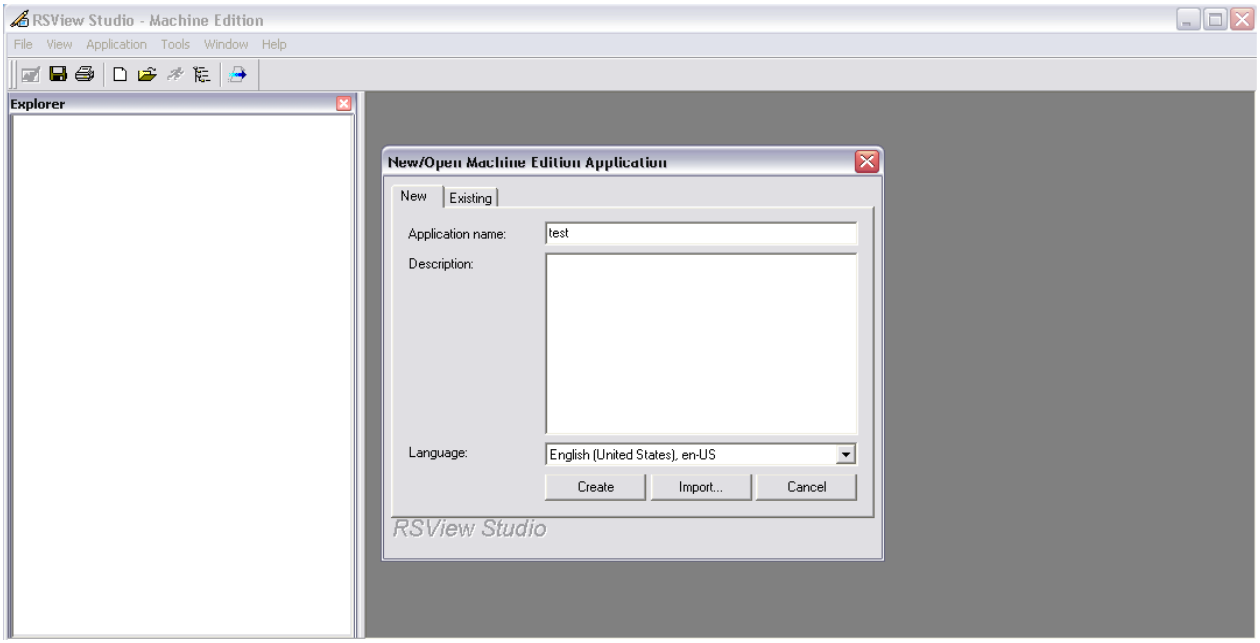
In the state table for the rectangle (button), change the number of states to 2 and tie the states to 'b', the coil in the example. When the button is pushed, the color changes from green to orange. The button now reflects the status of the coil 'b'.



We now have a button and indicator light similar to the ones used in the hard-wired system previously used. We have a button and illuminated indicator that may be used together or separately.

The RSView ME Stand-Alone Application

RSView Studio is the older A-B product. It resides on the PC's in NE 2350 but not in NE 2390. To open RSView Studio and gain access to RSView ME, open the program and click on the New tab. For Application name, the following example program used test.



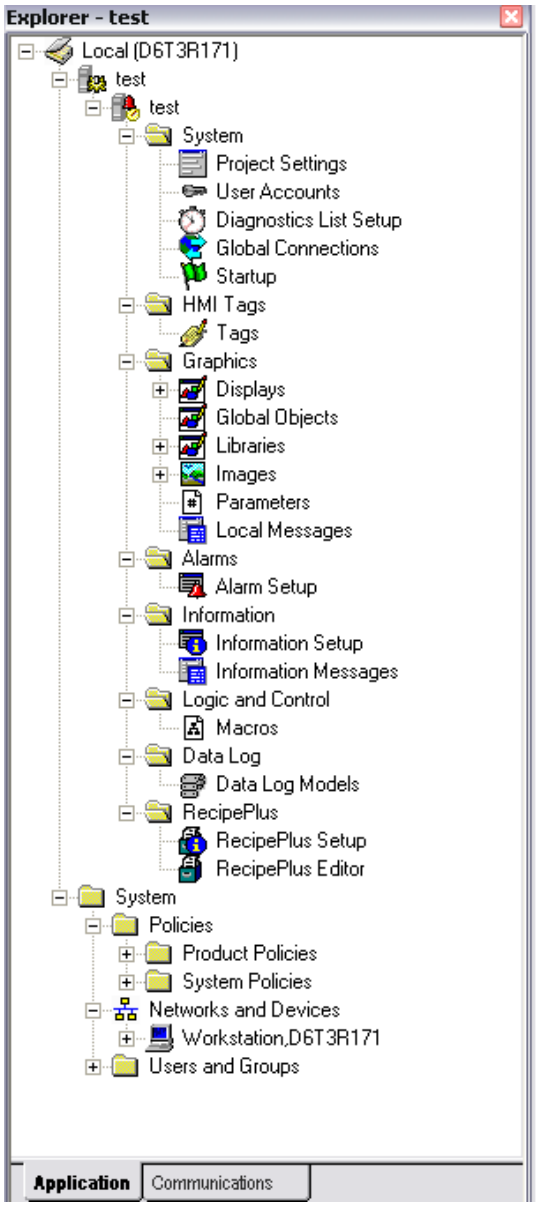
The A-B product uses RSLinx Enterprise. This means that RSLinx Enterprise has the ability to allow this software package to browse directly for the tag database and link existing tags and not requiring a tag created in the HMI to complement the tag in the PLC.

When starting a new application, select Objects from the main menu. Notice the types of objects

selectable. Under each selection are sub-menu selections. In the case of Push Buttons, the four sub-menu selections are:

Momentary
Maintained
Latched
Multistate
etc

The pushbutton is a good example of a type of object that can be used for a number of different operations. For example, momentary is the most used button with an exact simulation of a real button in which the operator pushes the button and a '1' appears in the data address of the device. When the button is released, the button returns to a '0'. As simple as this is, it is a very profound device in that the device continues to write a '1' to the data tag until the device de-activates and the device writes a '0'. Other button types have various other characteristics and these characteristics allow the programmer flexibility in programming of the various functions surrounding the logic of the pushbutton. More will be discussed later regarding some of the button types.



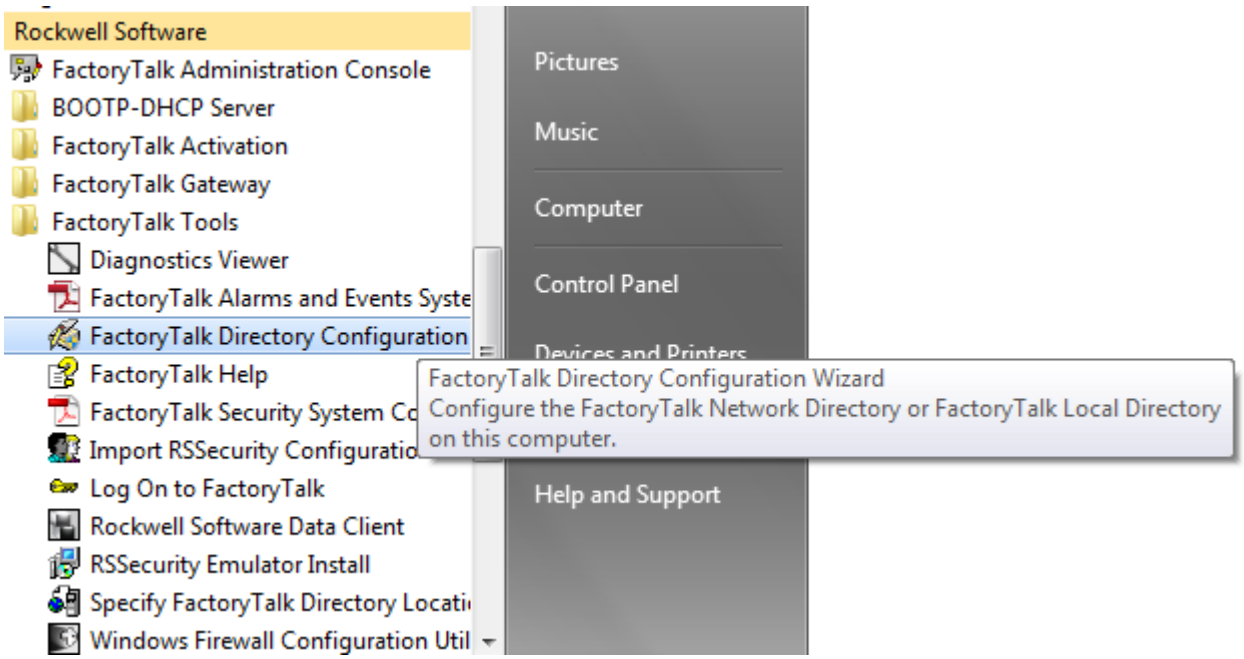
RSView Machine Edition (ME) is designed for the machine-level HMI and supports operator interface solutions for the monitoring and controlling of individual machines or for small processes.

The system tree at left shows the graphical application and is organized by area. These include:

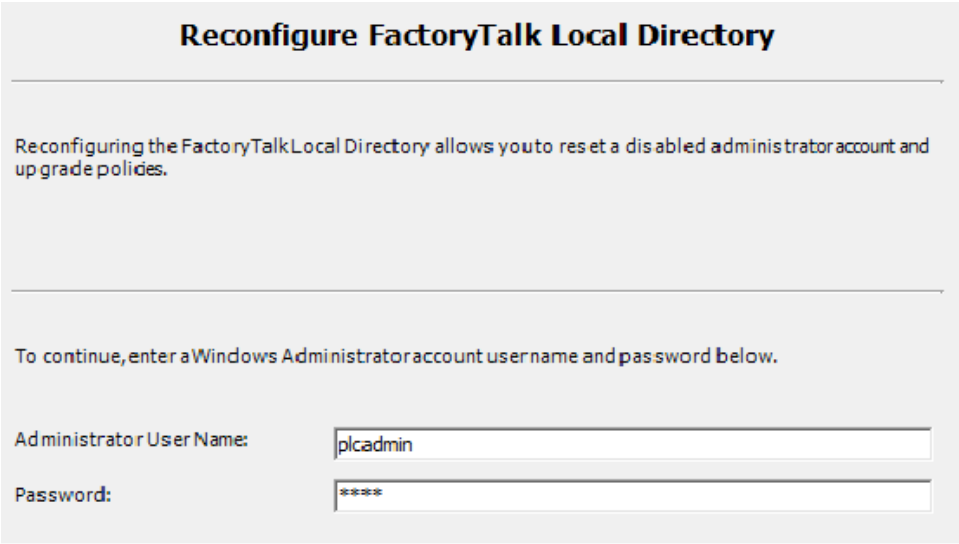
System
HMI Tags
Graphics
Alarms
Information
Logic and Control
Data Log
RecipePlus

For the simple assignments of this text, the user can simply add a display and create the appropriate graphics on the form. To test or run the form requires a link to a PLC. The procedure for linking to a PLC is included next. The tags created for the PLC are used in the graphic and any button or indicator will reference the PLC tag.

Remember that input and output bits used for a HMI are tags with internal addresses, not hard-wired inputs or outputs. For A-B, the device should have a tag referring to an internal location. For Siemens, the address should have an M prefix.



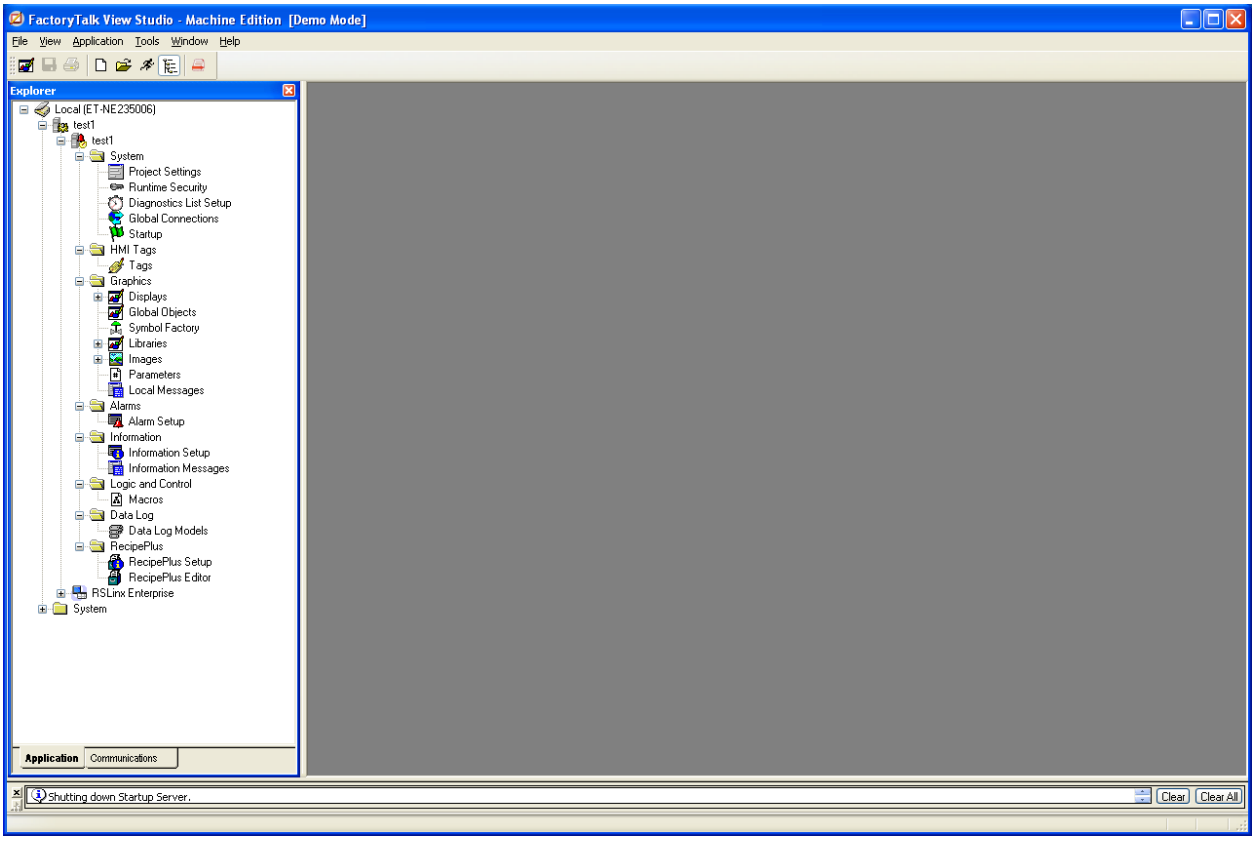
Then, configure the local directory per administration sign-in configuration of your pc.



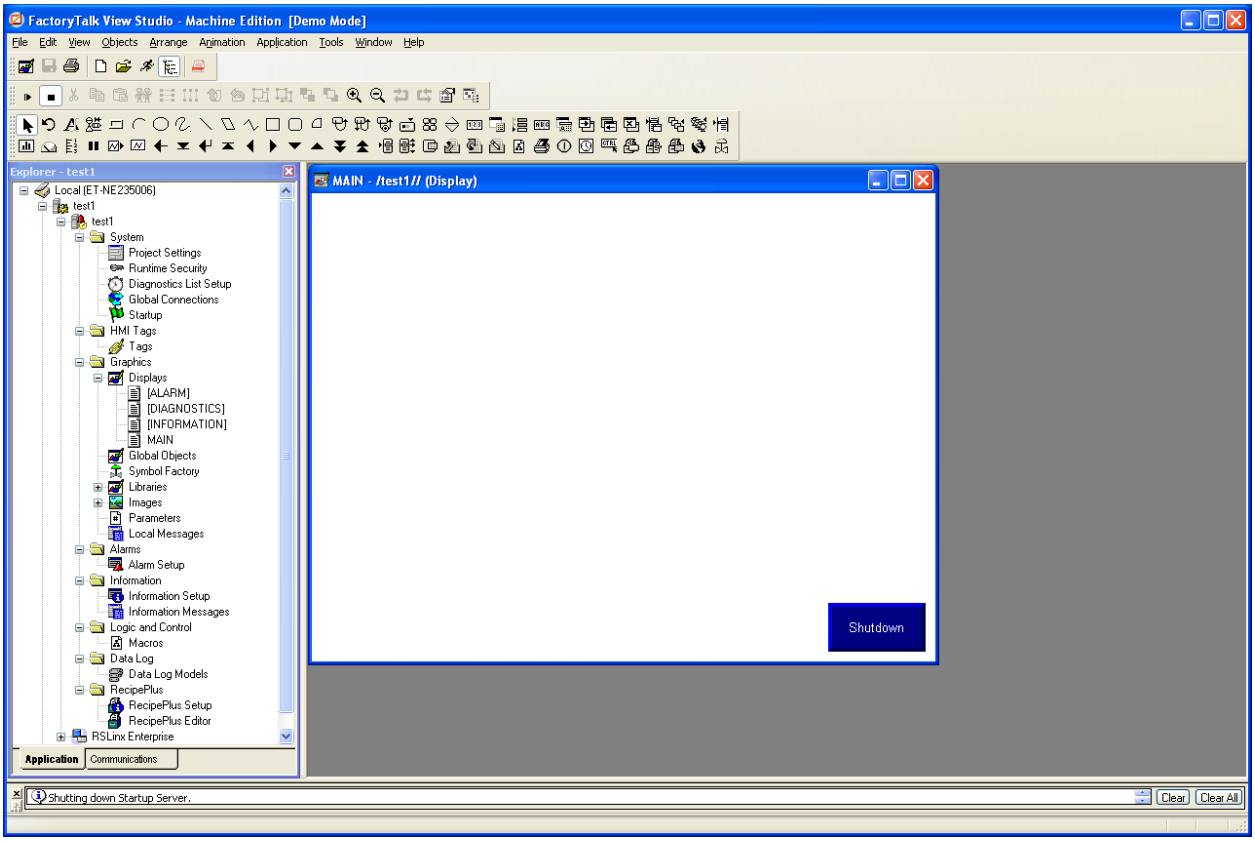
After this configeration is complete, you may proceed with your HMI application and connect the HMI screens to a PLC via RSView Enterprise as described above.

Creation of tags is accomplished either in the program tag or controller tag areas. Either may be accessed by FactoryTalk. The type of tag must be considered since numeric tags must be scaled and configured for proper display. Boolean tags must also be configured properly.

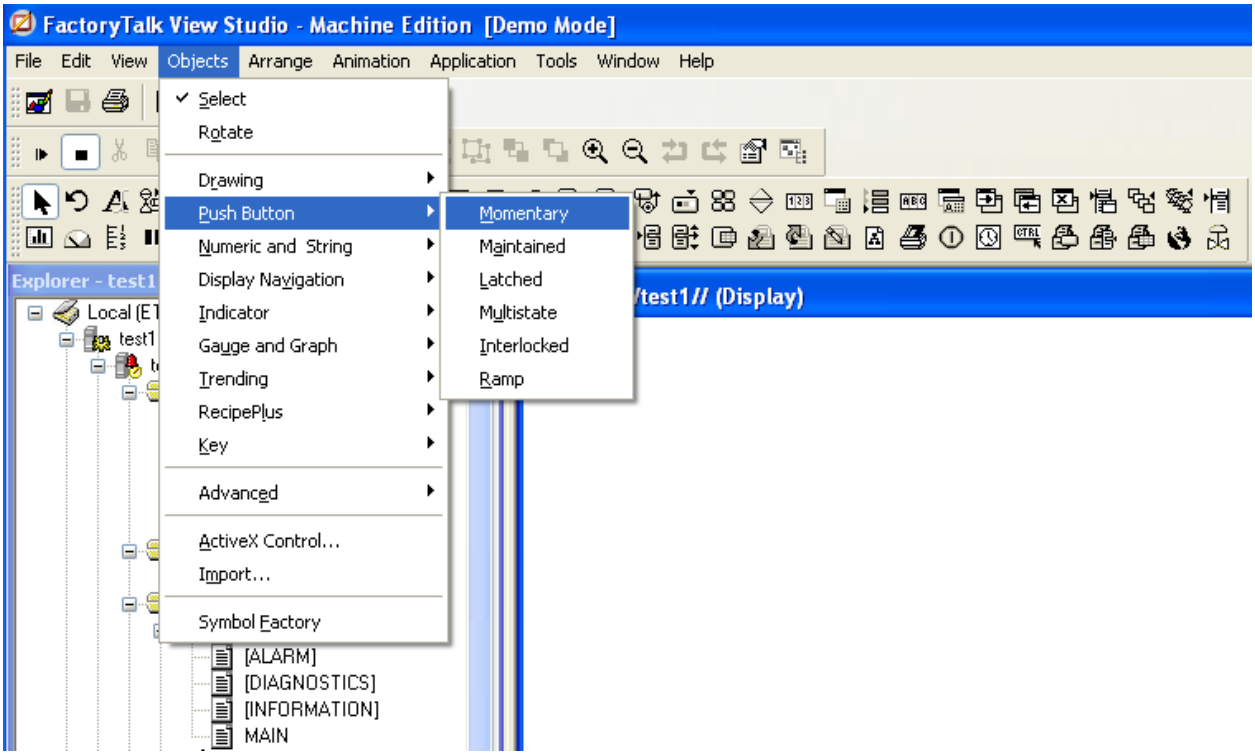
The project is begun with the following screen present. The project name is ‘test1’.



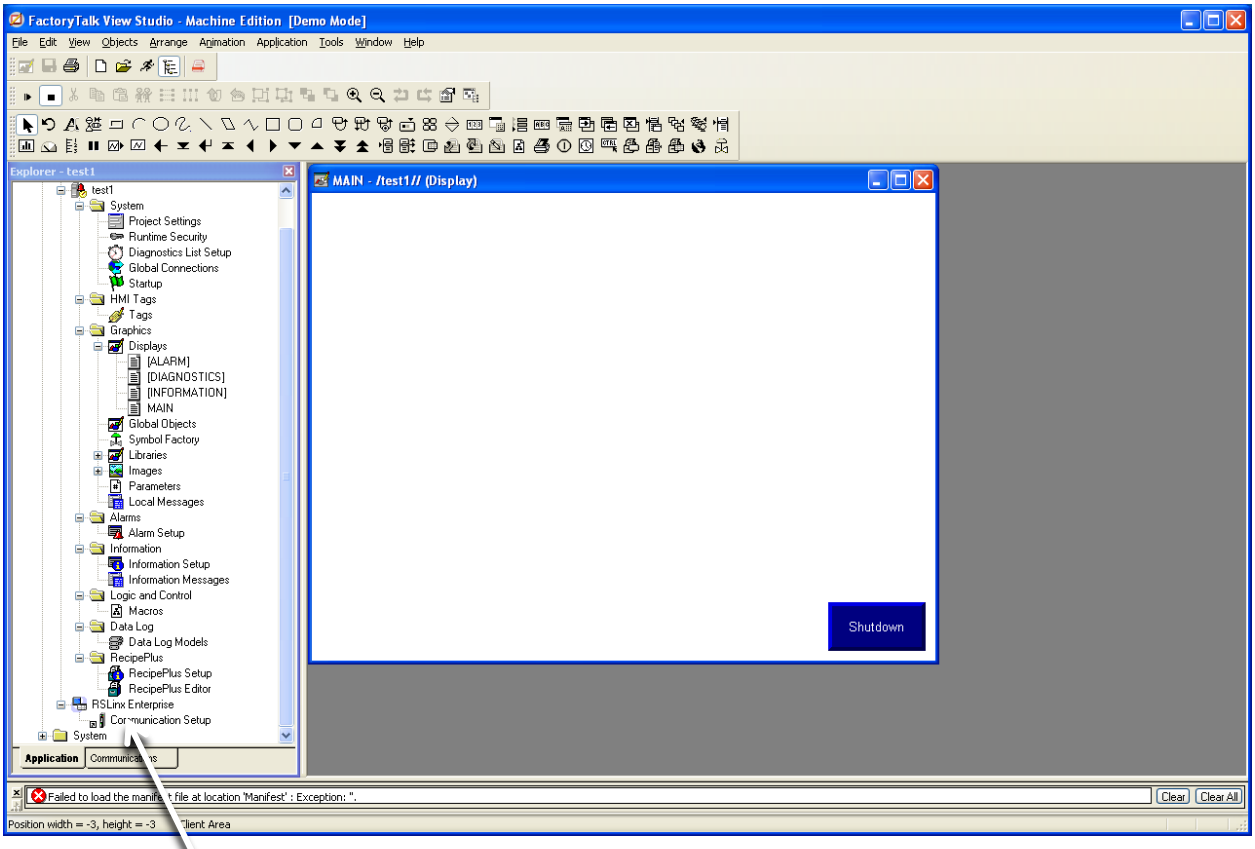
First Screen with RSView Studio



Selecting the Main Display

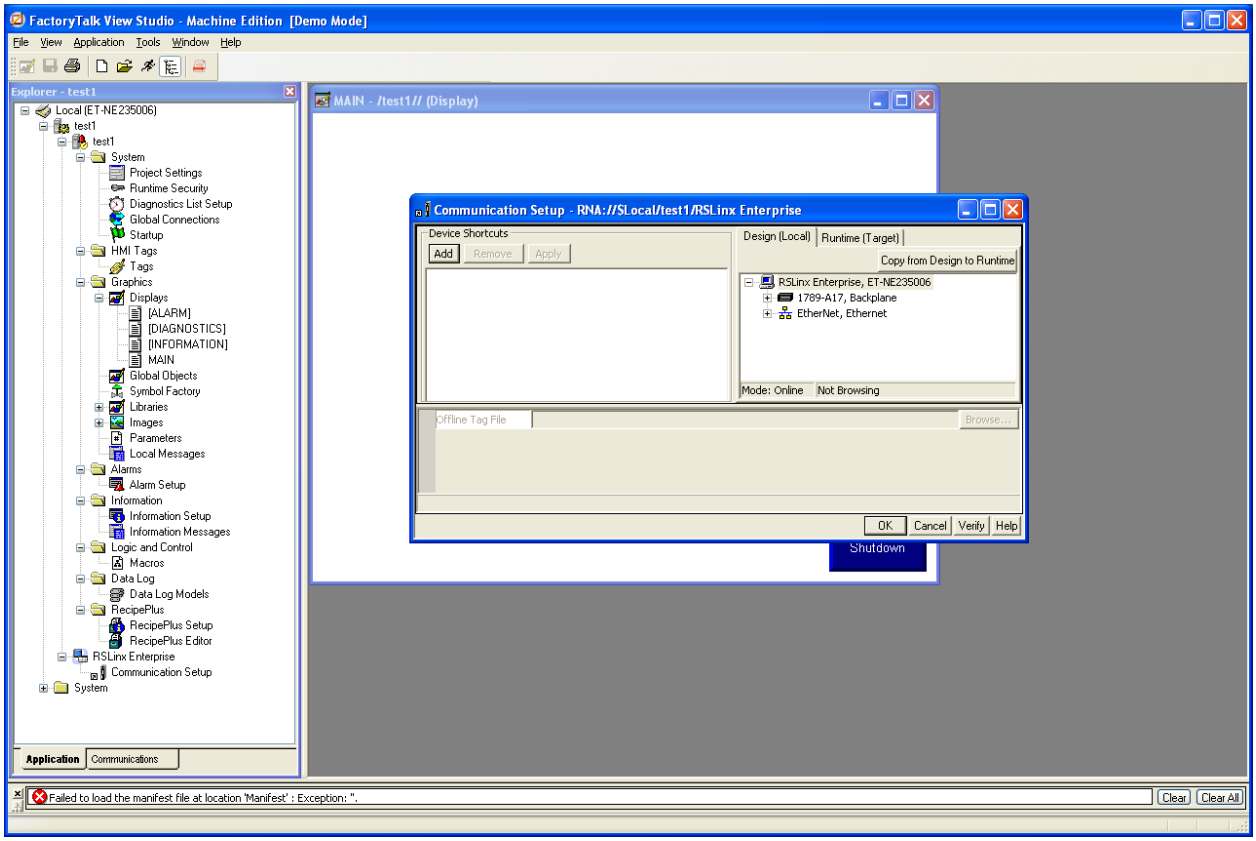


Building a Momentary Push Button

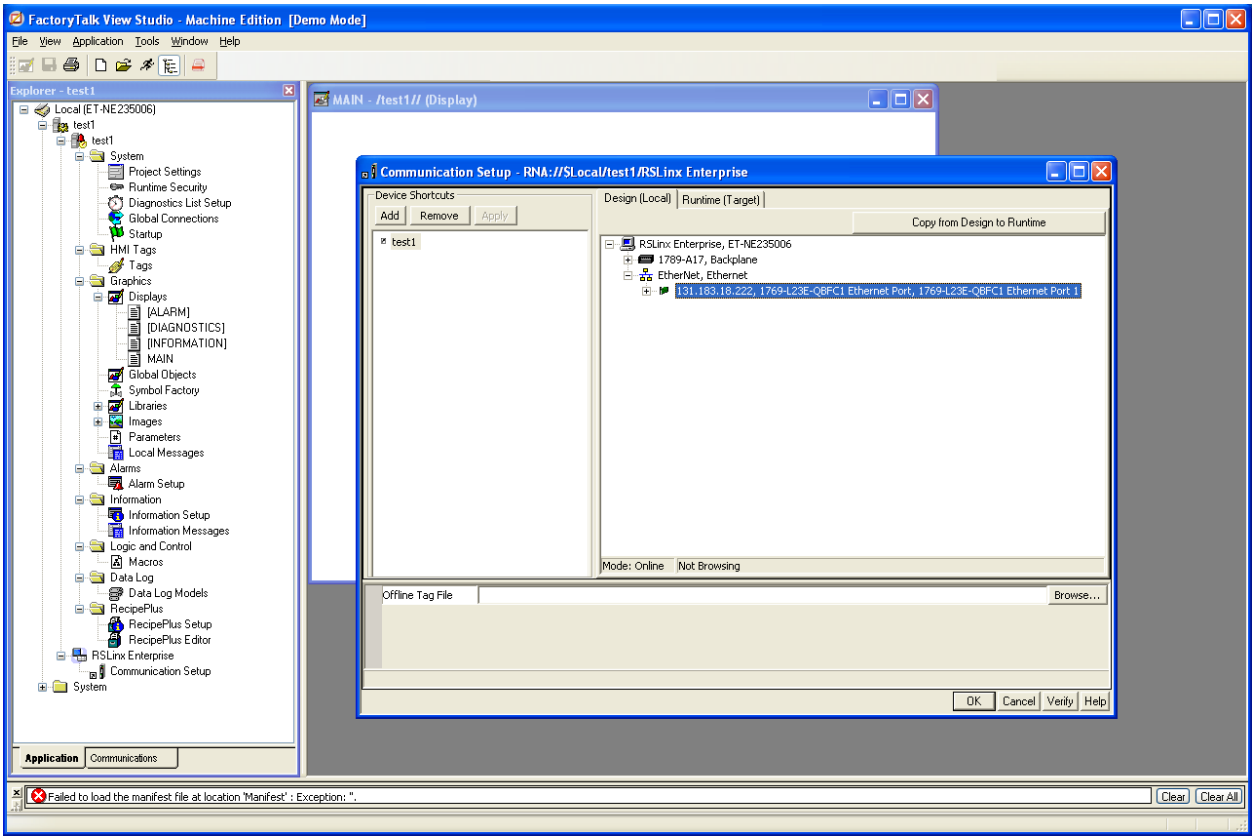


Establishing Communication with RSLinx Enterprise

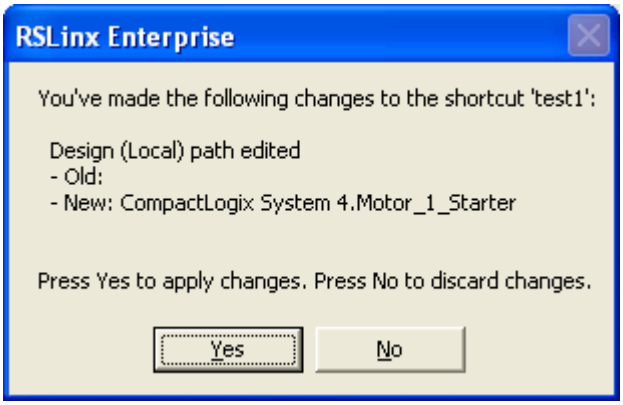
After Selecting Communication Setup under RSLinx Enterprise



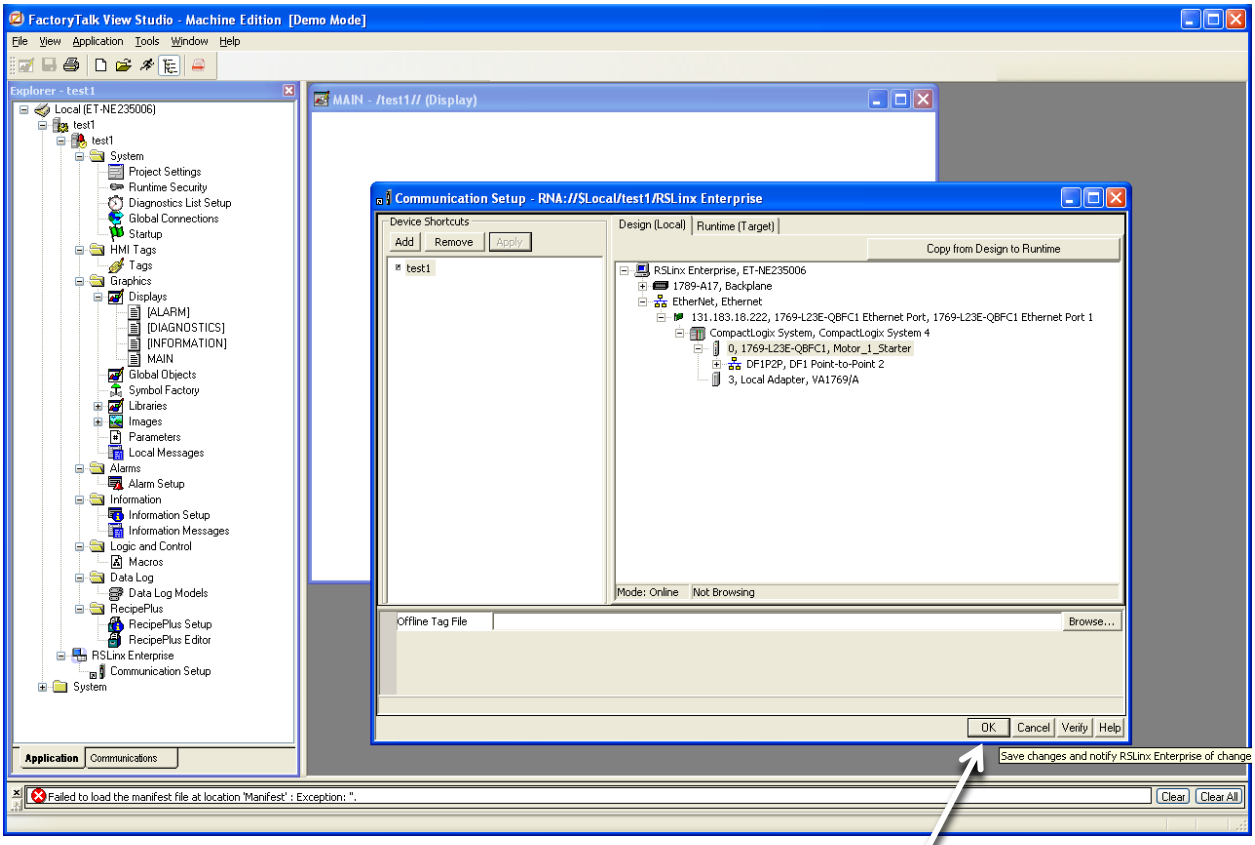
Click ‘Add’ for Device Shortcut, Enter Name ‘test1’



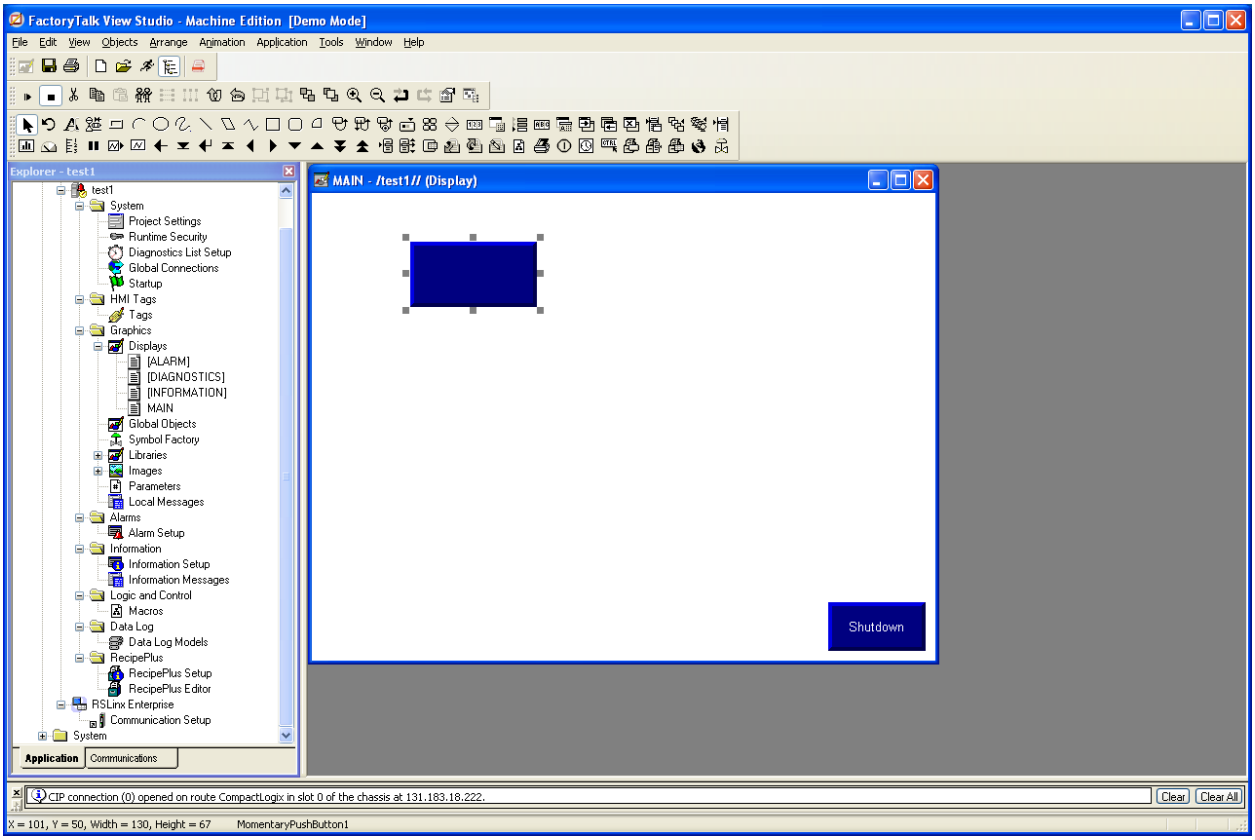
Highlight the Ethernet Device – PLC
When **APPLY IS HIGHLIGHTED - CLICK IT**



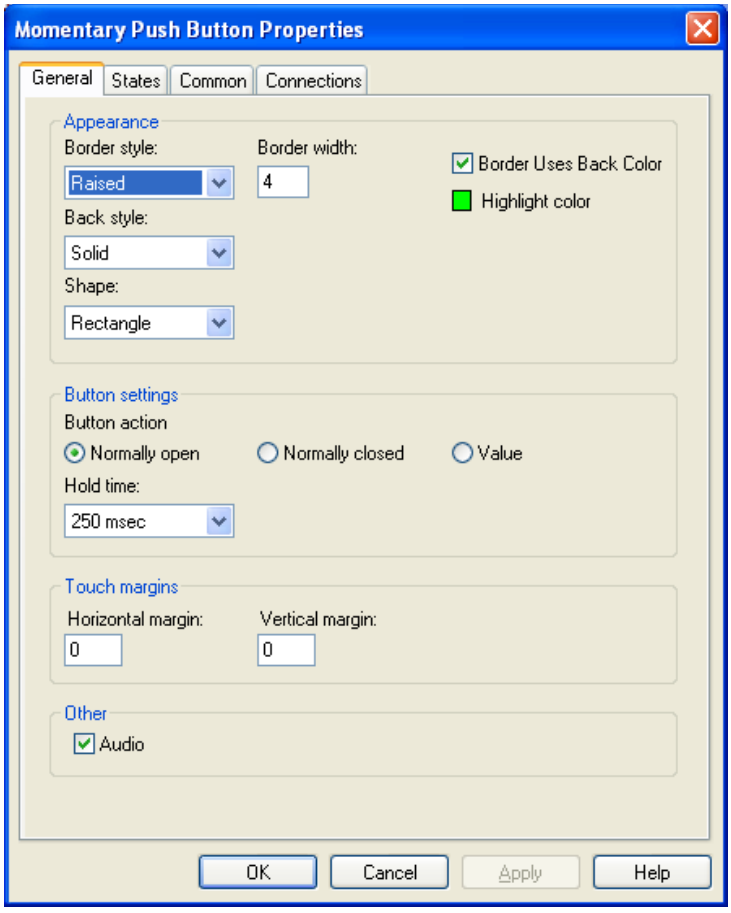
The Local Path is Determined for the Application



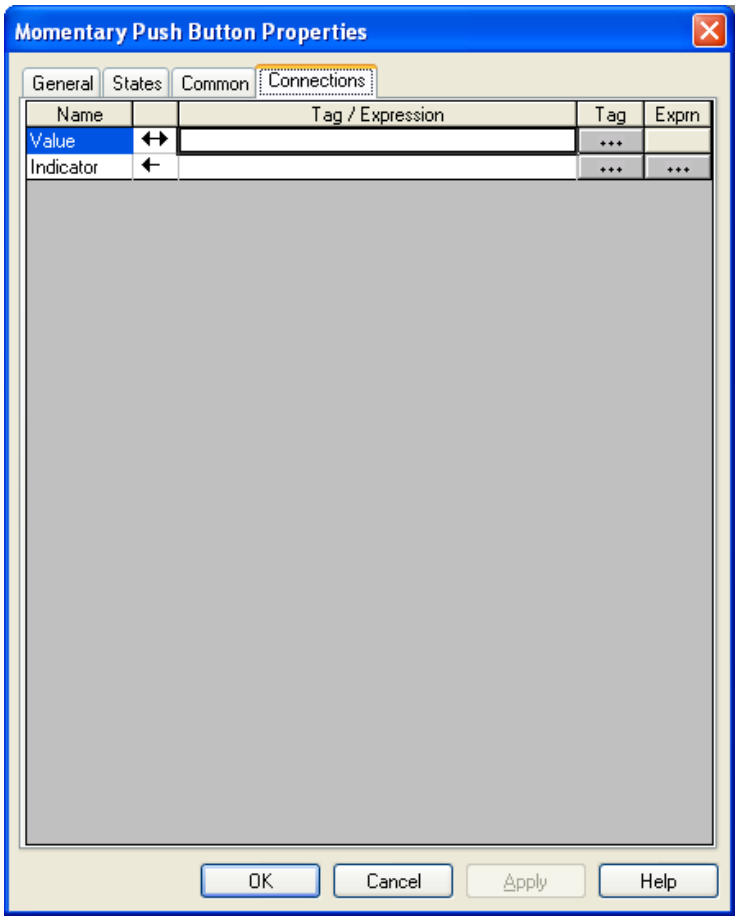
The Local Path is set with ‘OK’



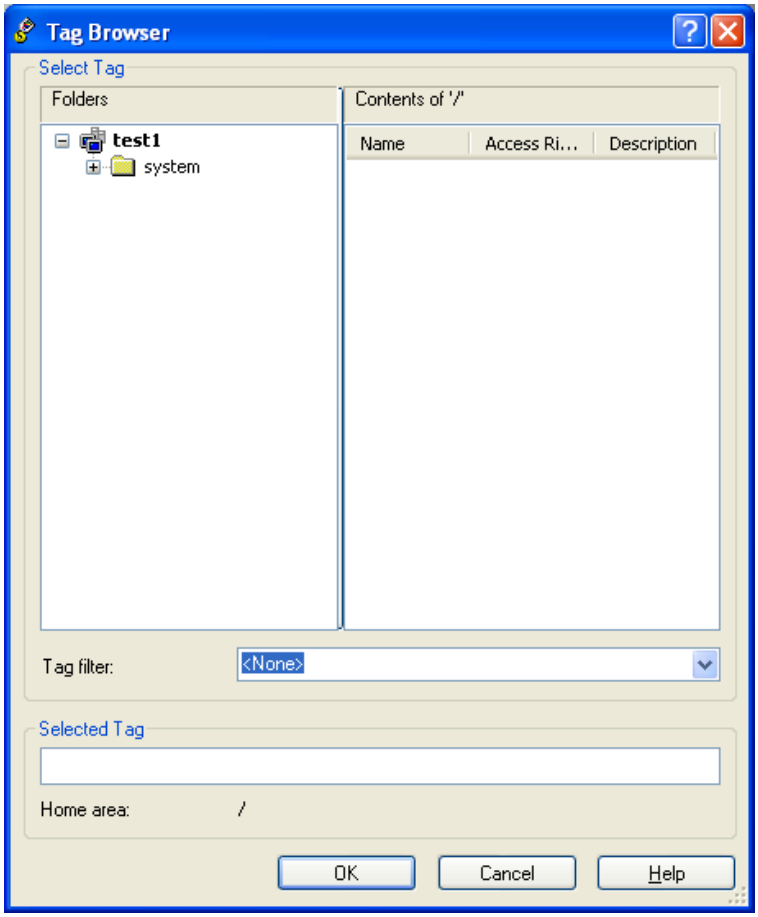
Back at the Ranch (I mean Button)



Working with the Button – Tab 1 - General

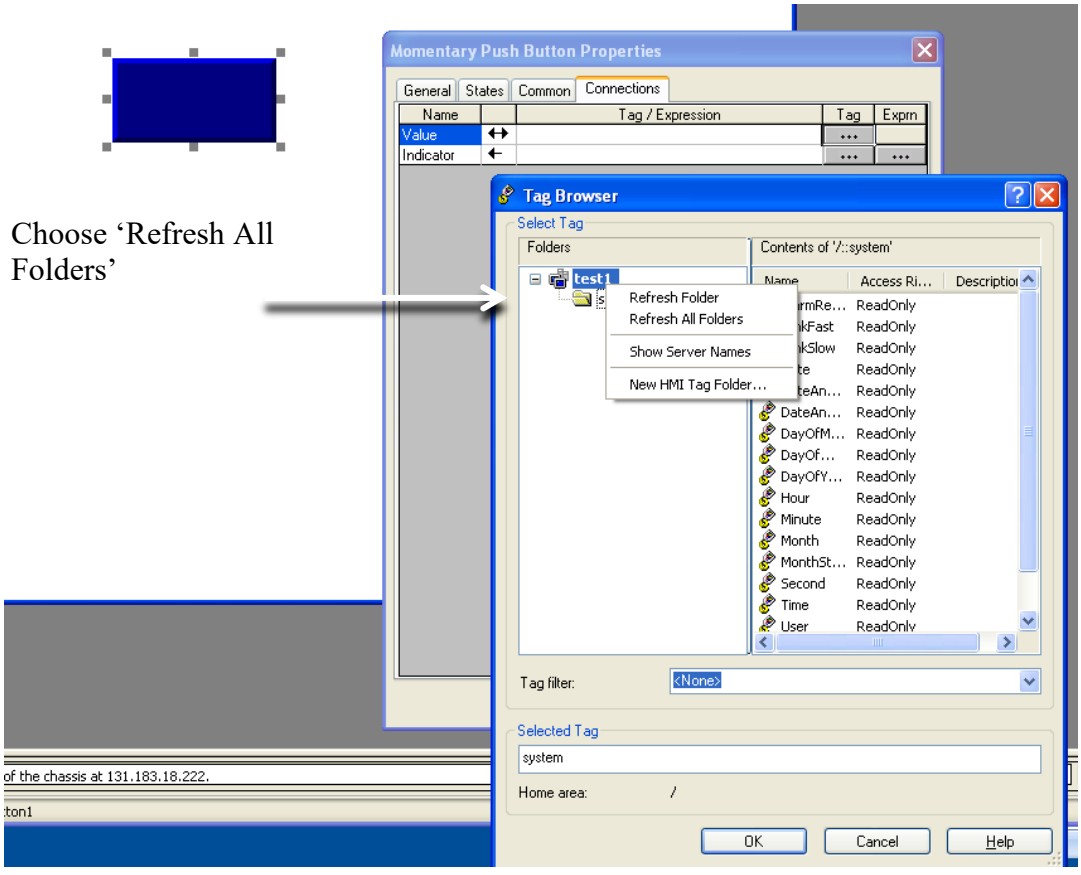


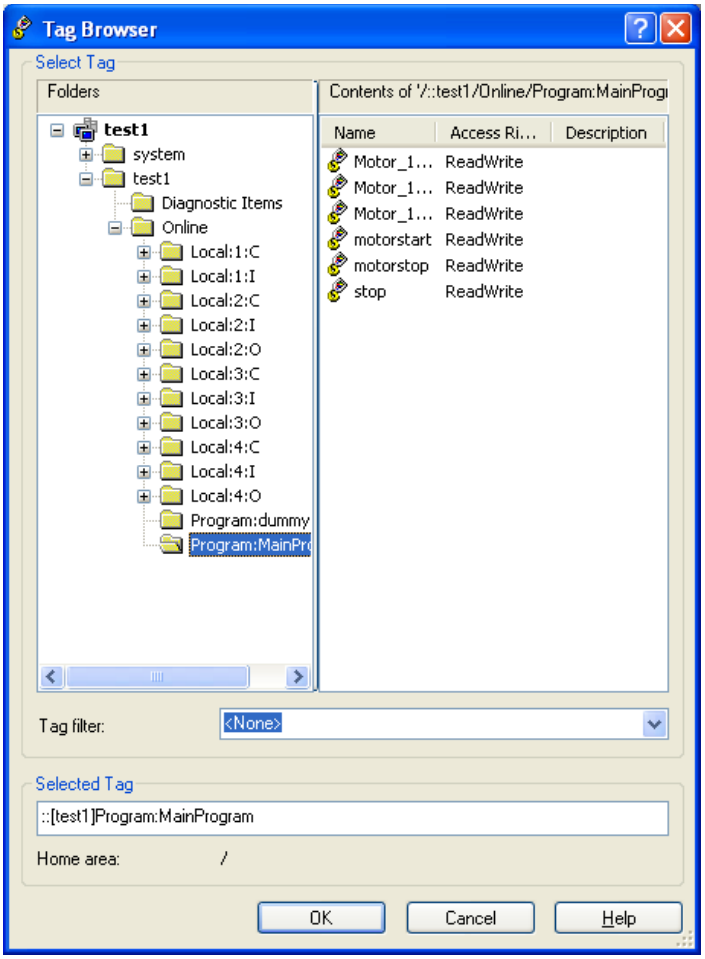
Establishing the PLC Connection to the Button



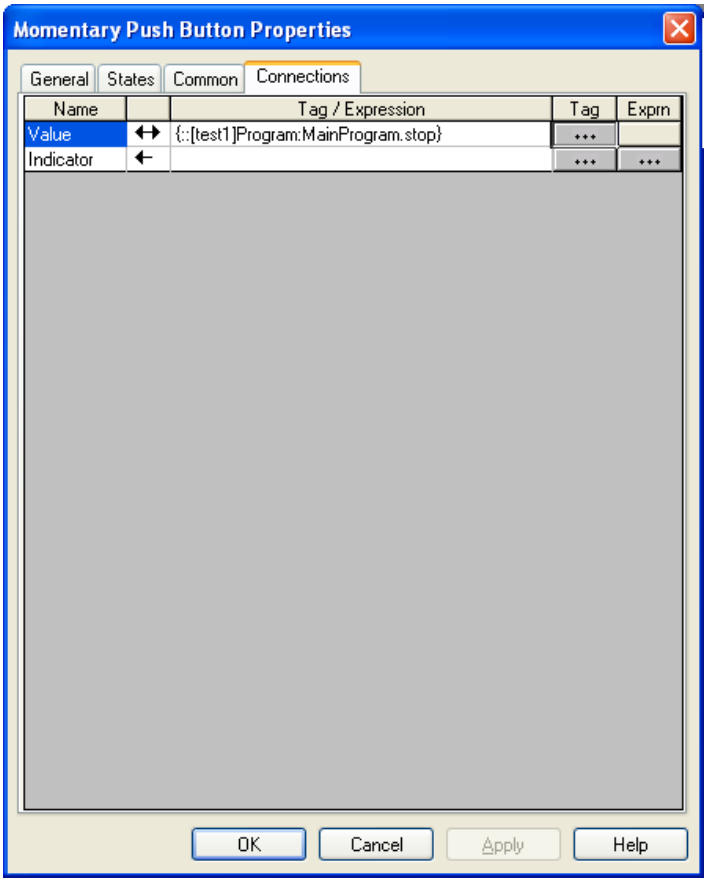
Click ‘Tag’ and the Link to the PLC ‘test1’ is Displayed

Choose ‘Refresh All Folders’

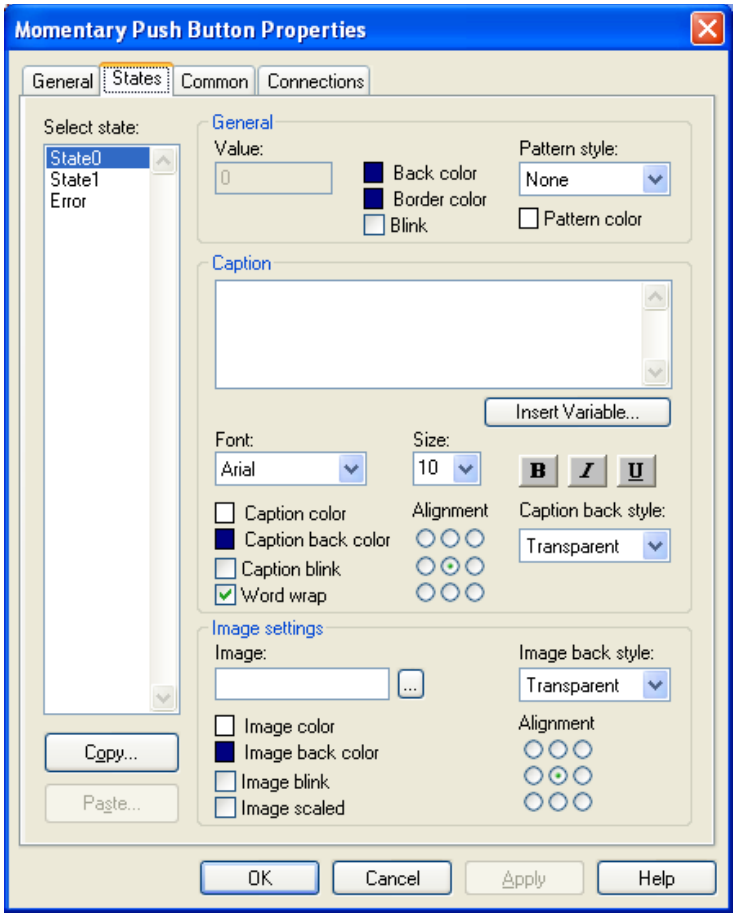




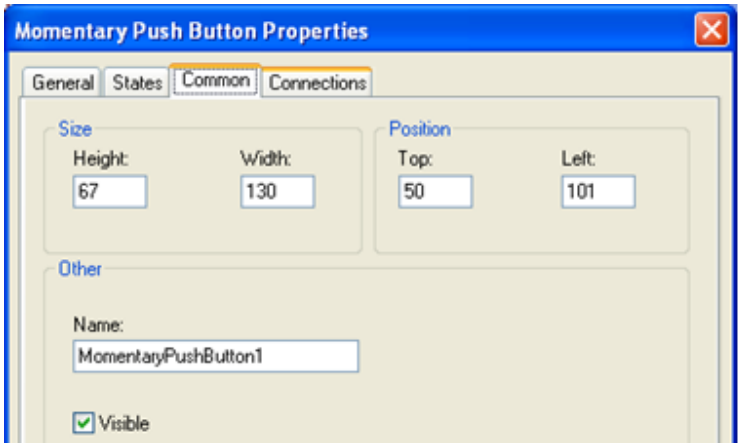
PLC Tags are Shown in the Online Tags



The Tag shows the Tag/Expression with the PLC Tag

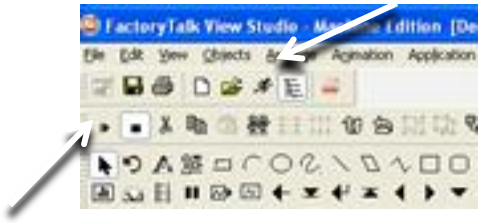


Color and Captions Modified under States Tab



The Common Tab Controls Size, Position and Name of the Button

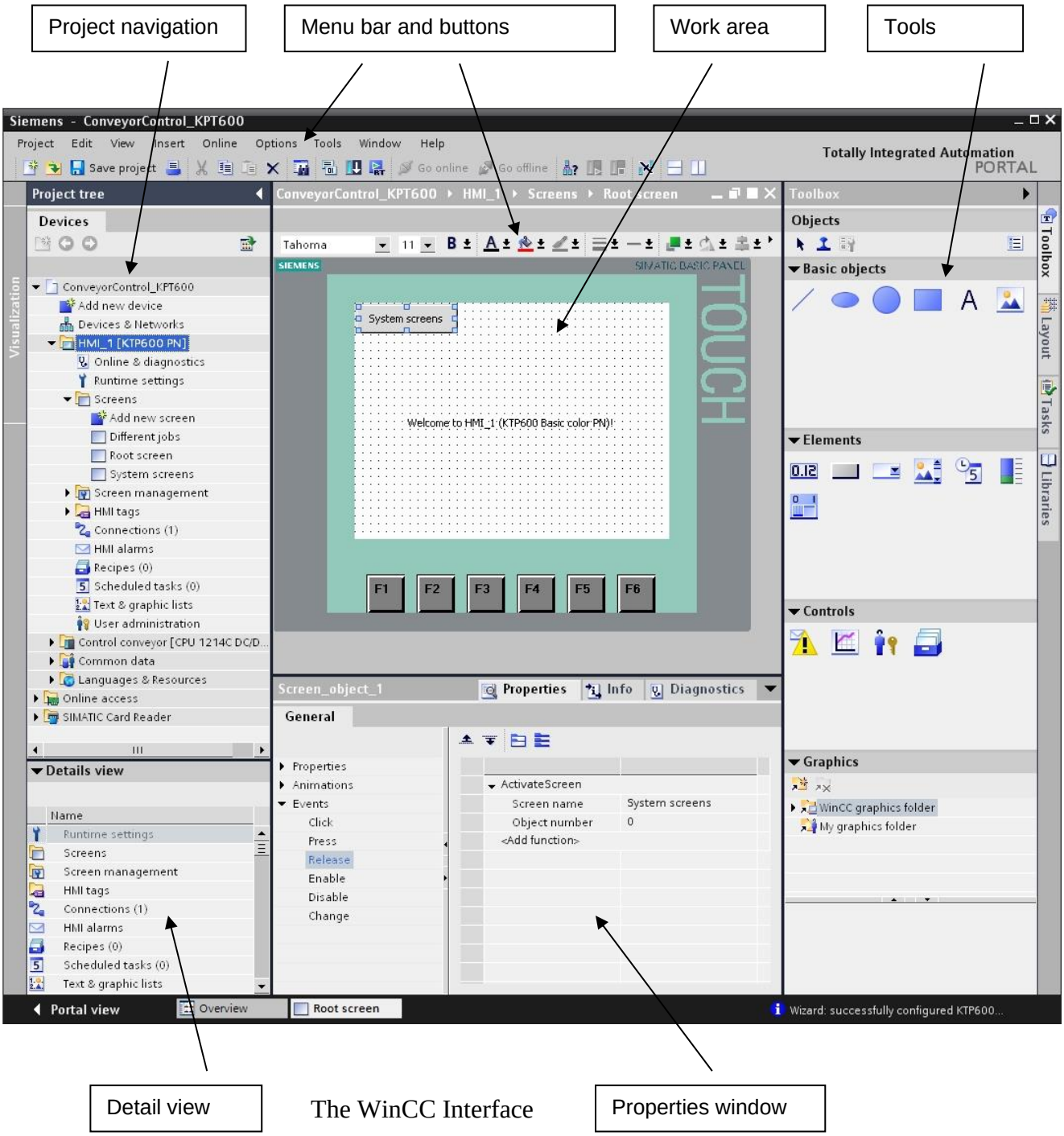
It is time to try the button with the connection to the PLC. Run the display by choosing the triangle in the command line. Test the screen with the button in the off and on position. View the bit in the PLC online program.



Use the triangle for testing single screens. To run all screens together in local mode, run the little man. There may be problems with this operation as it invokes a program called 'KEP' that may interfere with other applications. Be careful when running the little man.

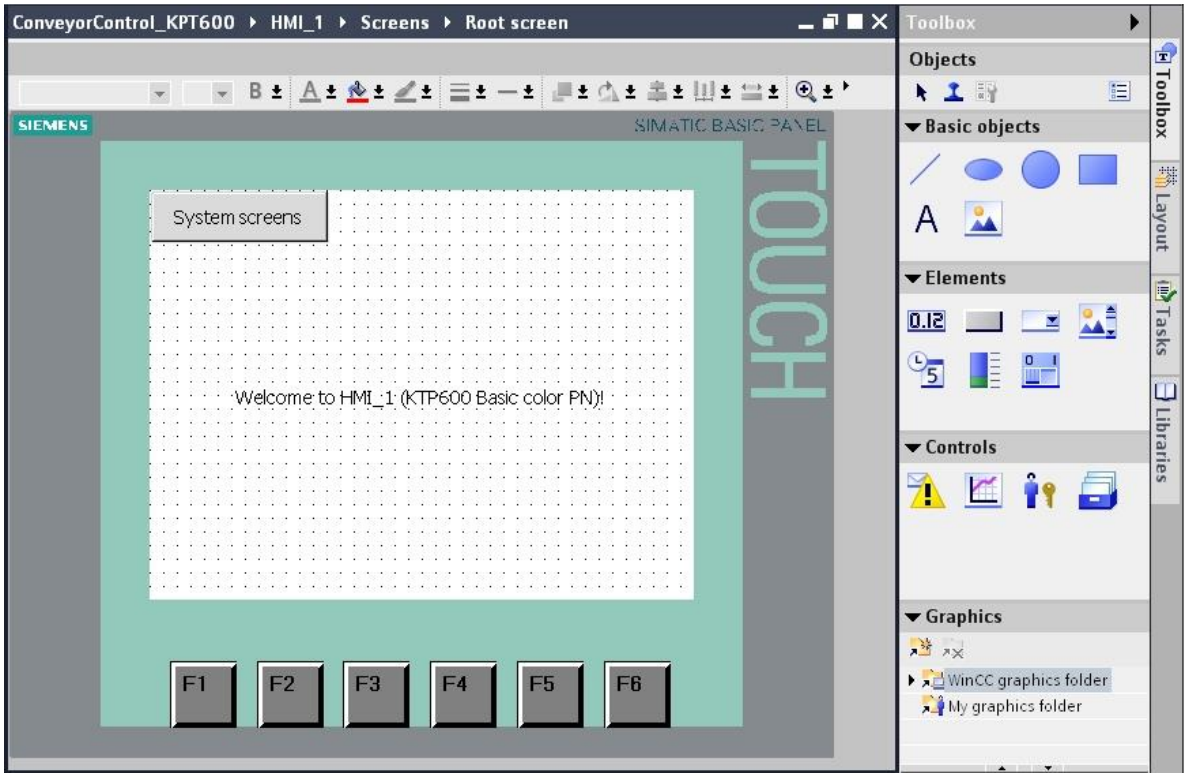
While building a screen or series of screens requires more instruction, we leave the A-B software to discuss Siemens' HMI interface.

Siemens Win CC
WinCC flexible Interface



Work Area

In the work area we edit the objects of the project. All elements of WinCC flexible are arranged around the work area. In the work area, we edit the project data either in the form of tables (for example, variables), or graphically (for example, a process display). The upper part of the work area contains a symbol bar. Here, the font, the font color or functions such as Rotate, Align, etc. can be selected.



The Root Screen

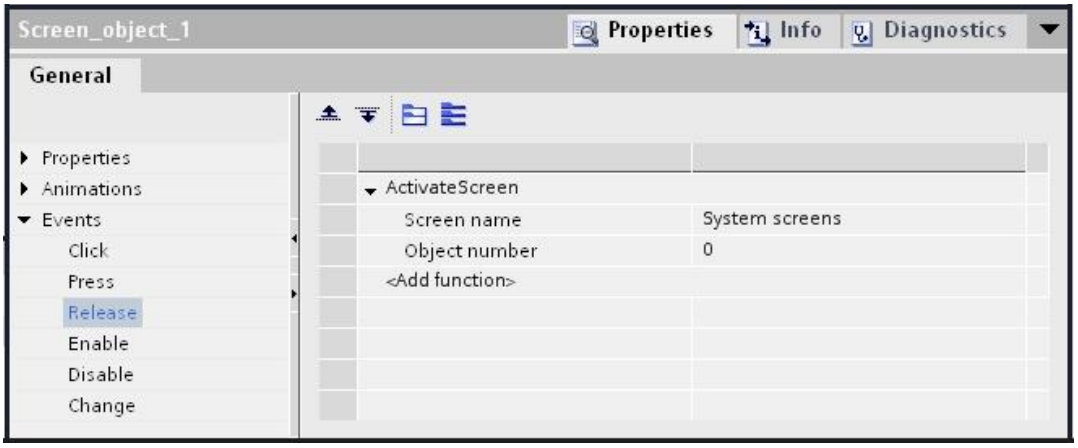
Tools

The tool window provides a selection of objects that you can insert in your screens; for example, graphic objects and operating elements. In addition, the tool window contains libraries with assembled library objects and collections of picture blocks. The objects are dragged and dropped into the work area.

Properties Window

The properties of objects are edited in the properties window; for example, the color of screen objects. The properties window is available only in certain editors. The properties of the selected object are displayed in the properties window, arranged according to categories. Value changes become effective as soon as an entry field is exited. If you are entering an invalid value, it is color-enhanced.

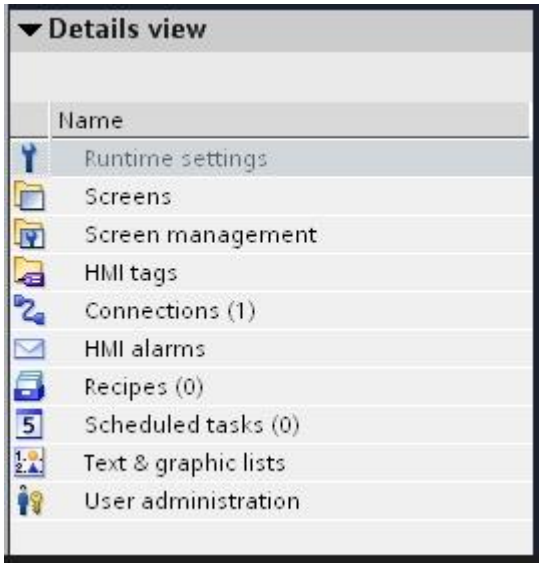
By using QuickInfo, information is provided about the valid value range, for example. In the properties window, animations and events of the selected object are configured also; here, for example, a screen change when releasing the button.



Dealing with Object Properties

Details View

In the Details view, additional details about the object highlighted in project navigation are displayed.



Details of the Object

Operating Screens and Connections

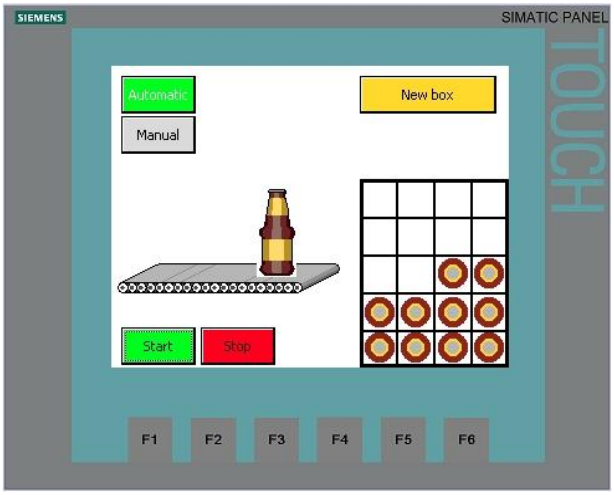
A screen can consist of static and dynamic components. Static components, such as text and graphs, are not updated by the control system. Dynamic components are connected to the control system and visualize current values from the control system's memory. Visualization can be in the form of alphanumerical displays, curves and bars. Inputs at the operator panel that are written to the memory of the control system are also dynamic components. They are interfaced with the control system by means of variables. Initially, we are only generating a screen for our conveyor control.

Root Screen or Start Screen

This screen was set up automatically and defined as start screen. Here, the entire plant is represented. Buttons can be used to do the following: switching the operating mode between

automatic and manual; starting and stopping the conveyor motor, and exchanging the box. The movement of the bottle on the conveyor belt and the fill level of the box are represented graphically.

Using button F6, we are jumping to the system screen:

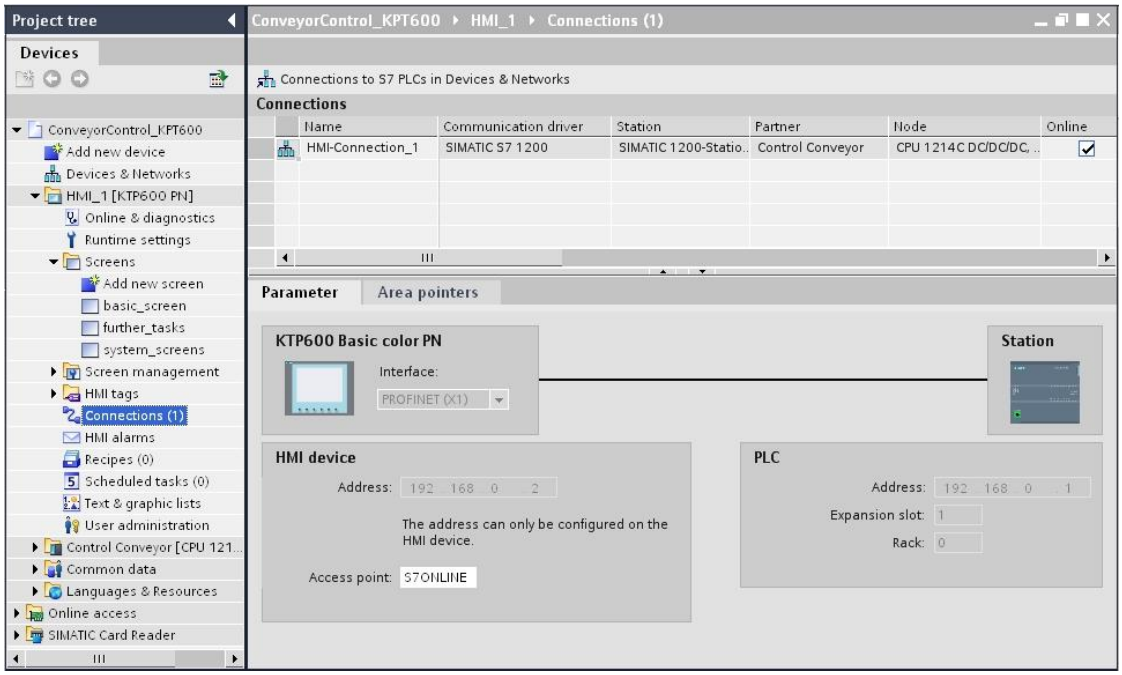


The System Screen

Connections to S7 Control Systems

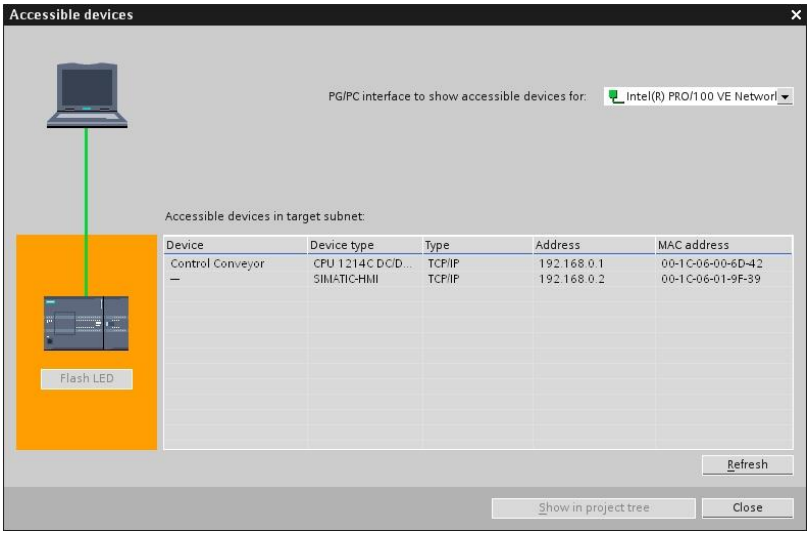
In the case of operator objects and display objects that access the process values of a control system, a connection to the control system has to be configured first. Here we specify how the panel communicates with the control system, and with which interface.

In Project navigation, click on Connections. Because of the settings in the hardware configuration, all parameters are already set.



Establishing the PLC Connection

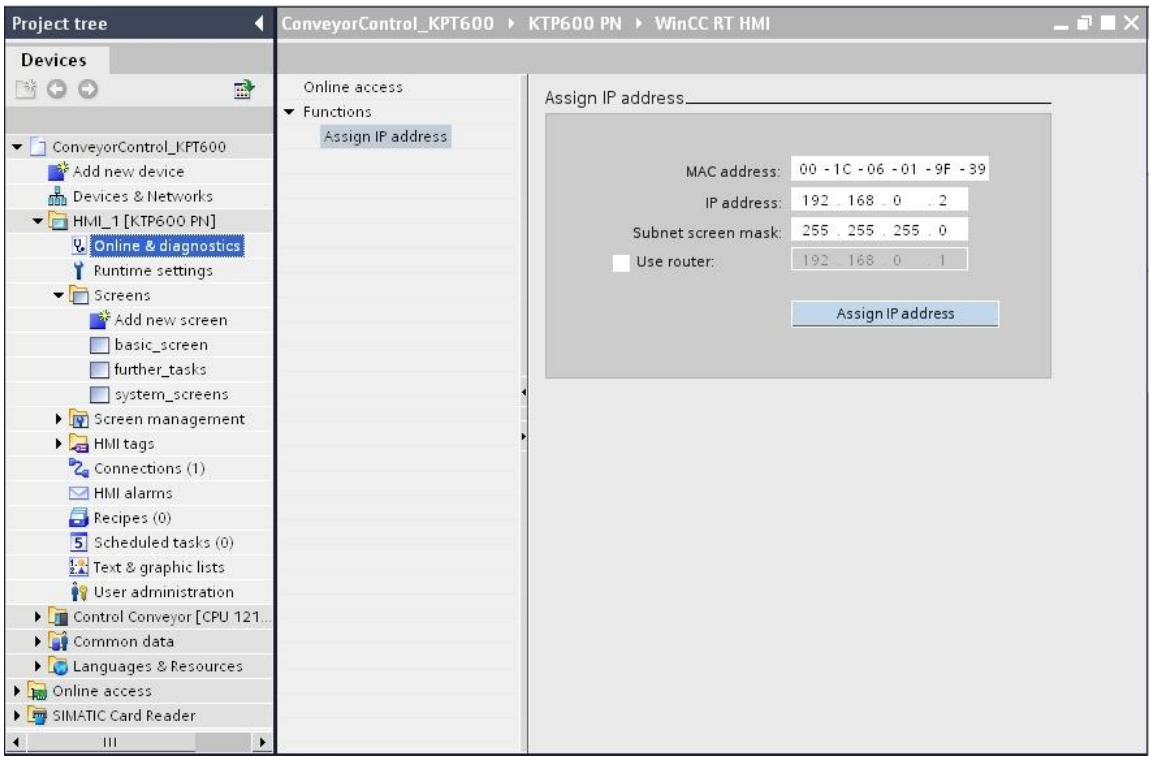
The IP address has to be assigned to the panel also. By means of Accessible devices, read out the panel's MAC address, or read it on the back of the panel.



Assigning the IP Address

Assigning the IP Address

After inputting the MAC address, the IP address can be assigned under Online & diagnostics. The panel has to be in the Transfer Mode in this case. Change the IP address to **192.168.0.5** for all applications.



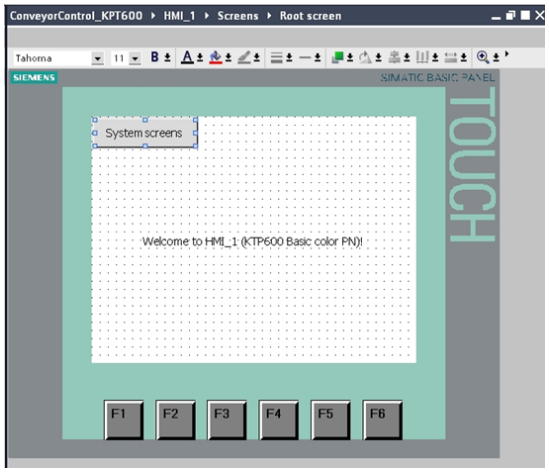
Note

The IP address can also be checked or entered on the panel in the system control under Control Panel at Profinet.

Configuring the Root Screen

Clicking on the button “System screens“ displays the system screen. We want to transfer the function of the button System screens to the function key F6.

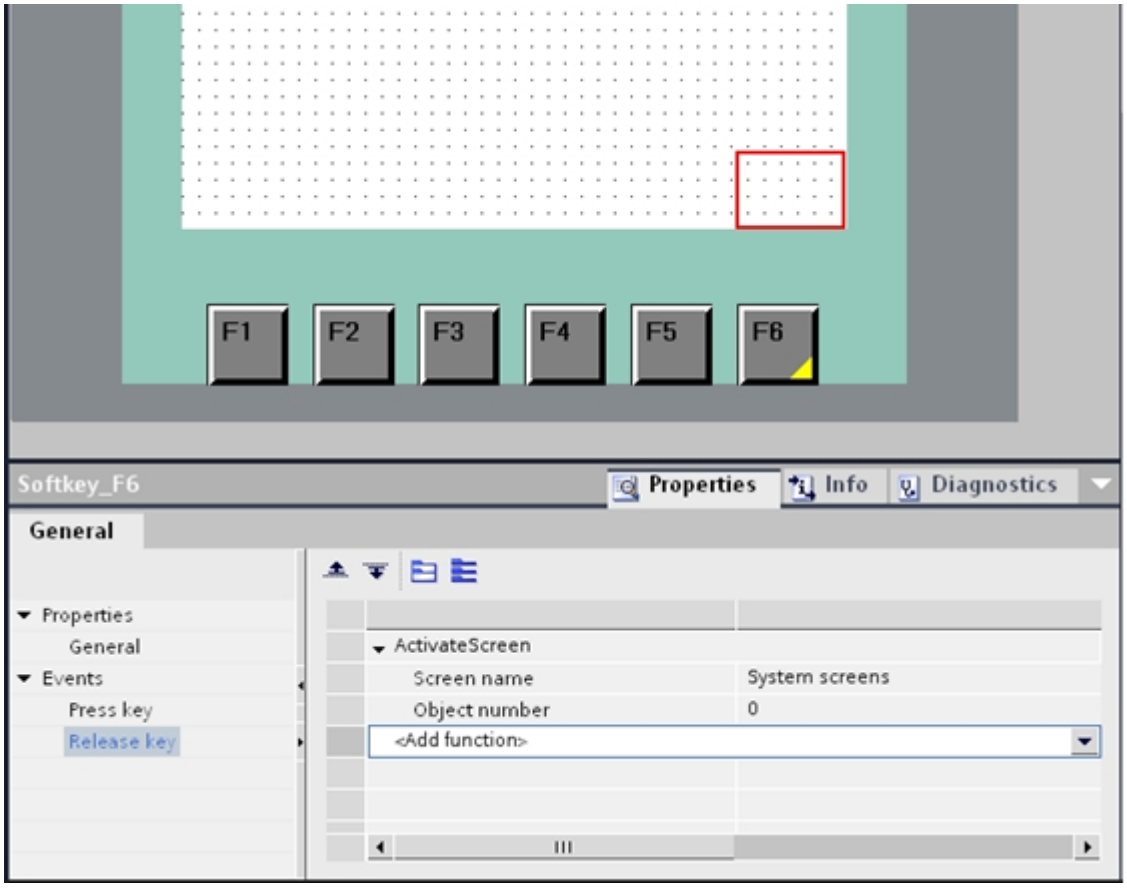
Select System screens and in the Properties window below copy the function Activate screen at Events Release.



A Start-Up or System Screen

Function Key F6

Select function key F6 and in the Properties window below, insert the function Activate screen at Events Release key. Then, delete the text field in the center, and delete or remove the button System screens.

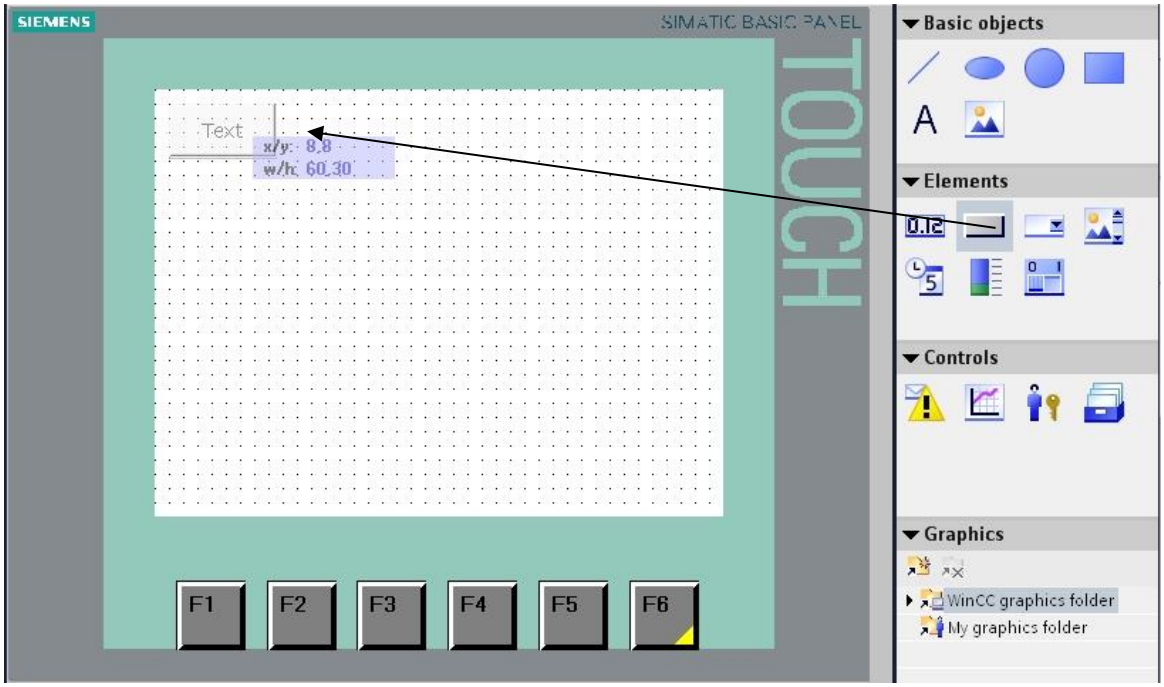


Defining a Function Key

The yellow corner on the F6 key indicates that the key is configured.

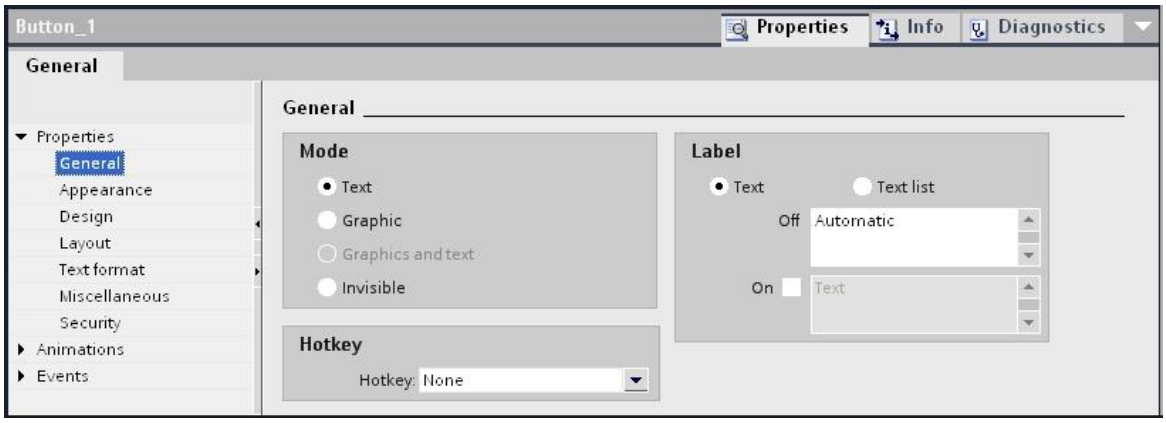
Configuring the Buttons Automatic and Manual

Drag a button into the work area of the root screen.

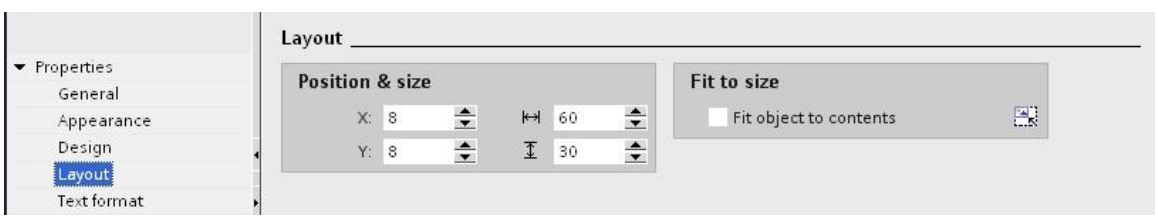


Configuring a Button

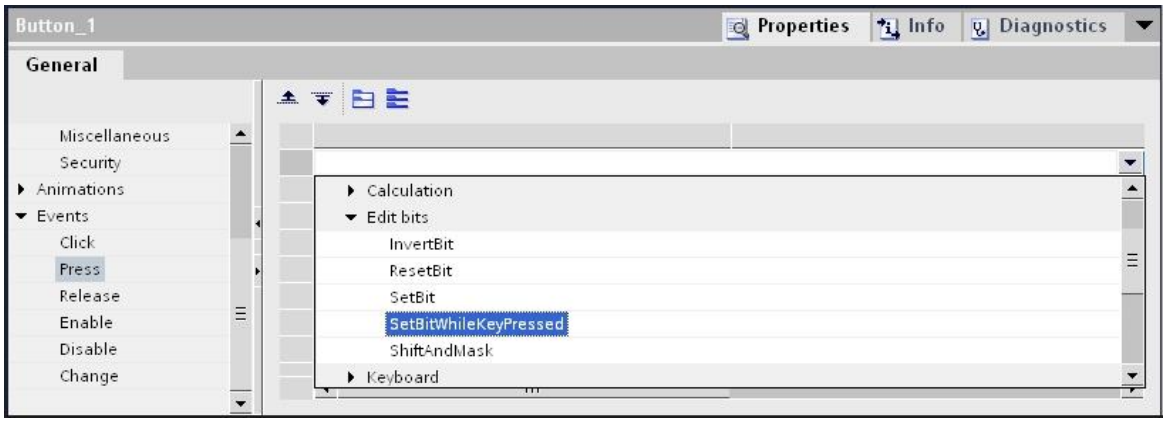
At Label, enter Automatic.
Caution! Don't press the input key; otherwise, a second line is generated.



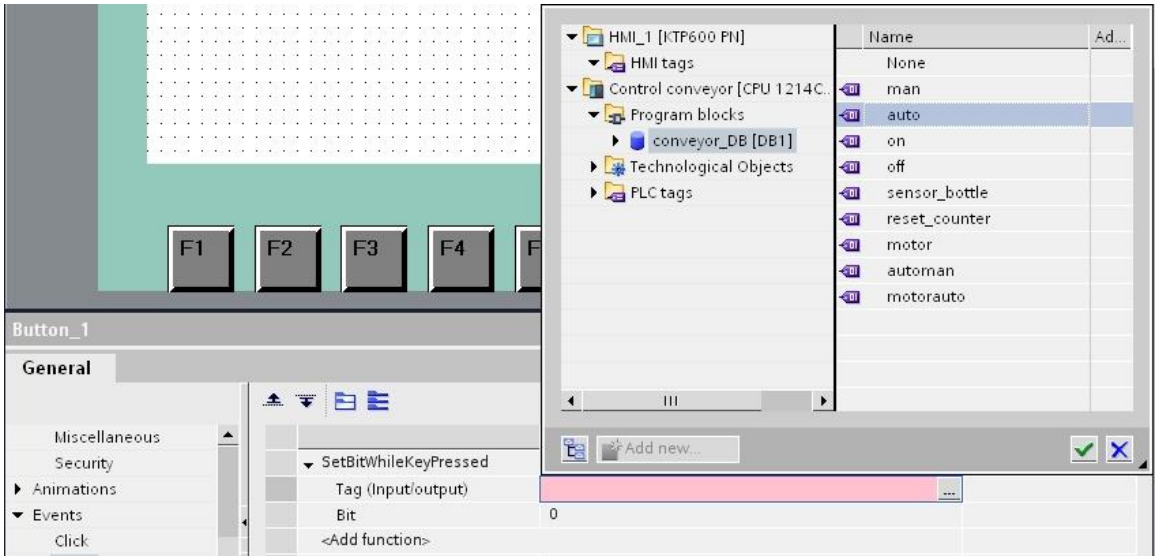
Under Layout, enter position and size.



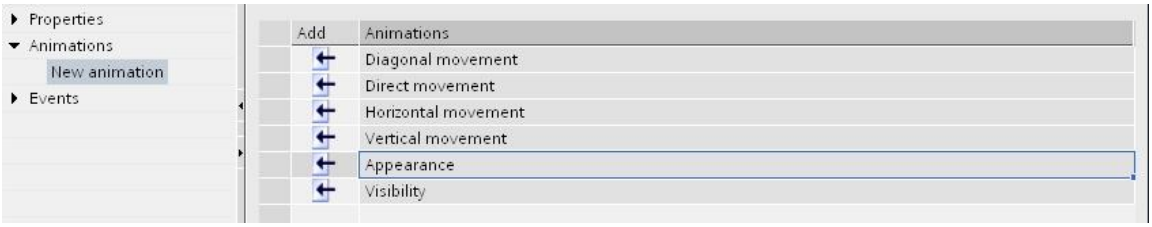
Under Events Press select the function SetBitWhileKeyPressed under Edit bits.



Then, click on the field Tags (input/output) and using "**…**" button, open the tag window. Here, it is also possible to access the interface declaration of data blocks. As tag, select auto from the Conveyor_DB [DB1].

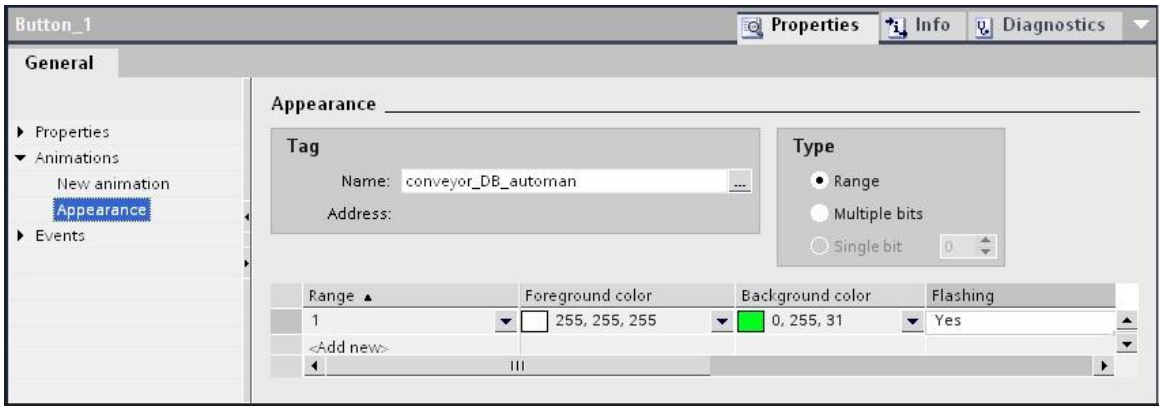


Now, the button is to flash in the automatic mode, and change color. With a double click, select under Animations\New animation Appearance.



As tag, select automan of Conveyor_DB [DB1].

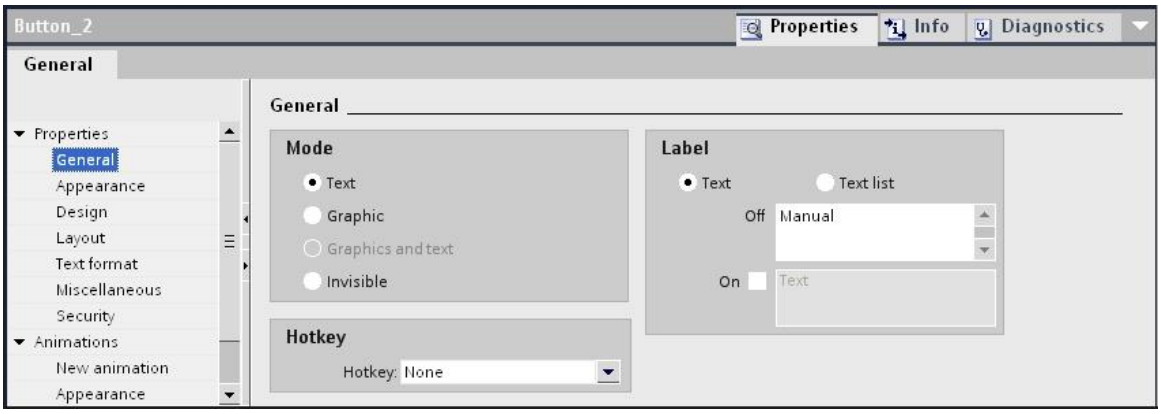
The button is to change color in the automatic mode; that means, when the variable automan has the value 1. For the color change to become visible, change the foreground color at Appearance to White and the background color to Green. At Flashing, say Yes.



Change Color of Button

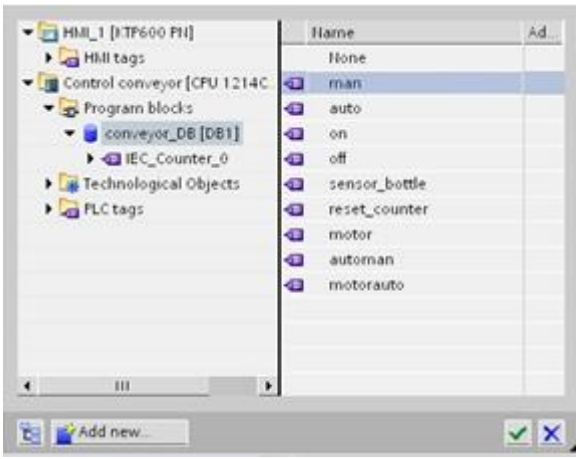
Copy and paste the button Automatic. Place the inserted button under the Automatic button.

At Label, enter Manual. **Caution!** Don't press the input key; otherwise, a second line is generated.



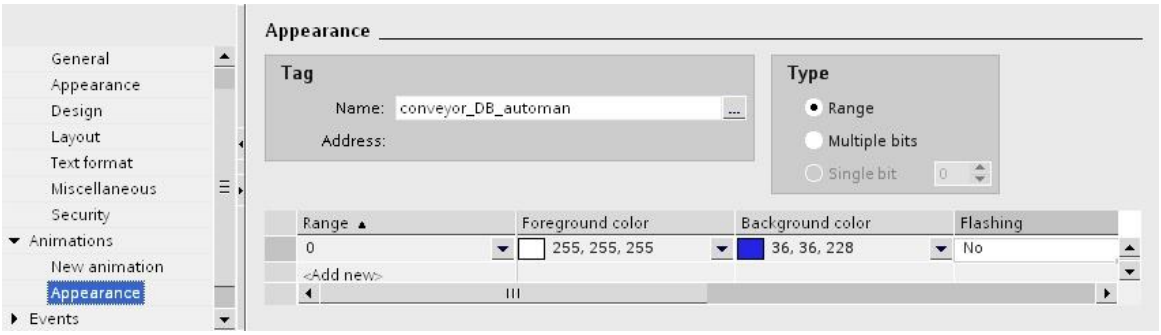
HMI Tag

Under Events Press, select man from Conveyor_DB [DB1] as tag. The variable has to be selected, because only then will the new HMI tag be generated.



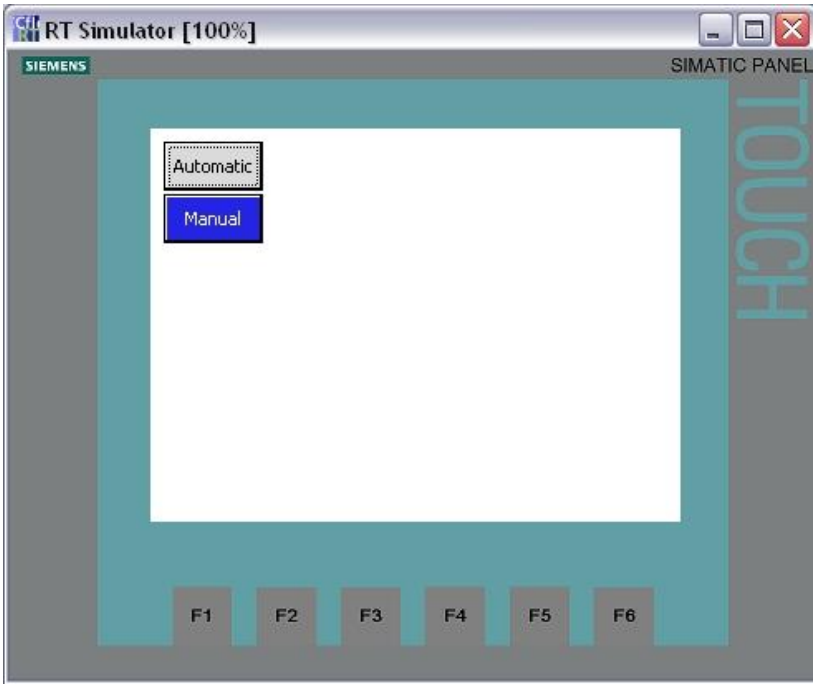
More Example
HMI Tags

The button is to change color in the manual mode; that means when the variable automan has the value 0. For the color change to become visible, change the foreground color at Appearance to White and the background color to Blue. Set Flashing to No.



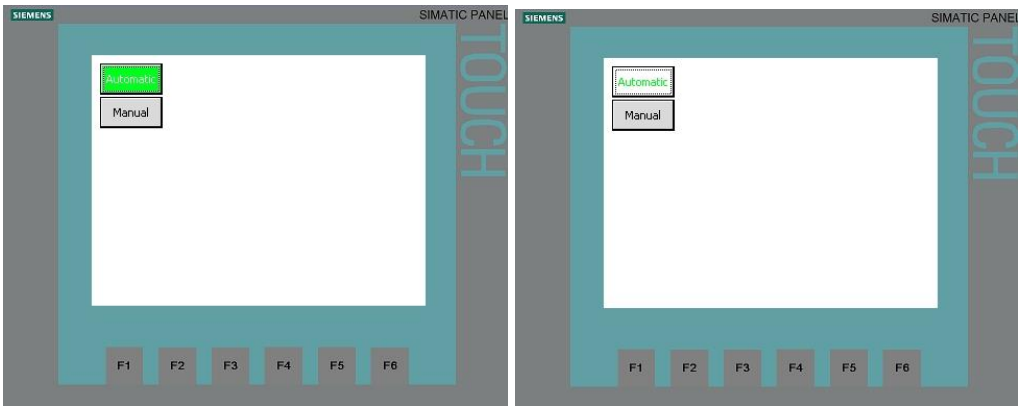
Button Appearance

Visualization is opened in the RT simulator.



Running the Simulator

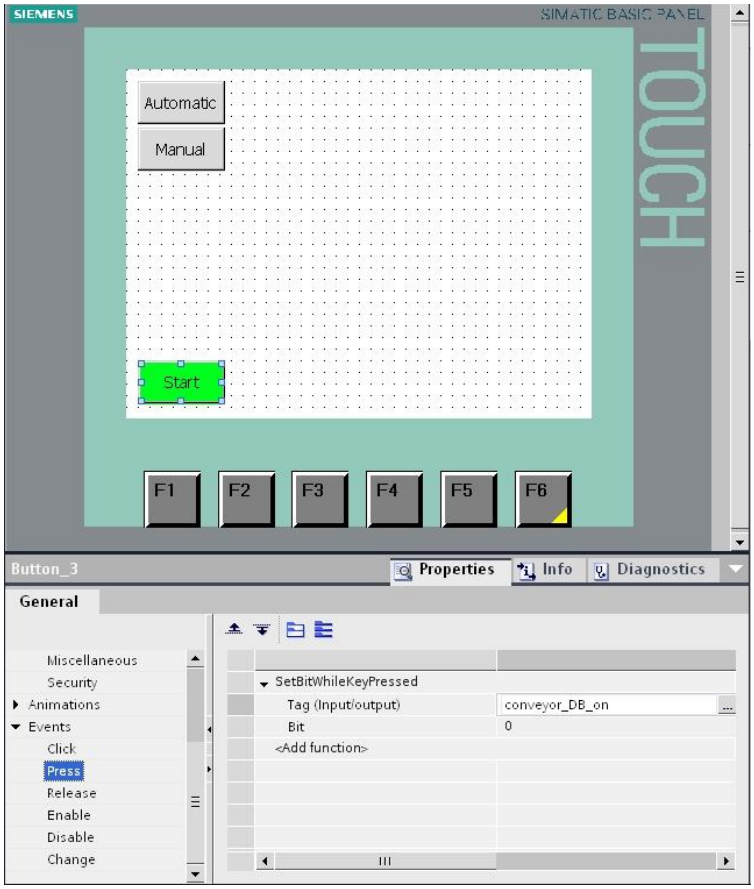
Test the project of the conveyor control.
The automatic or manual mode is now pre-selected on the panel.



Start and Stop Buttons

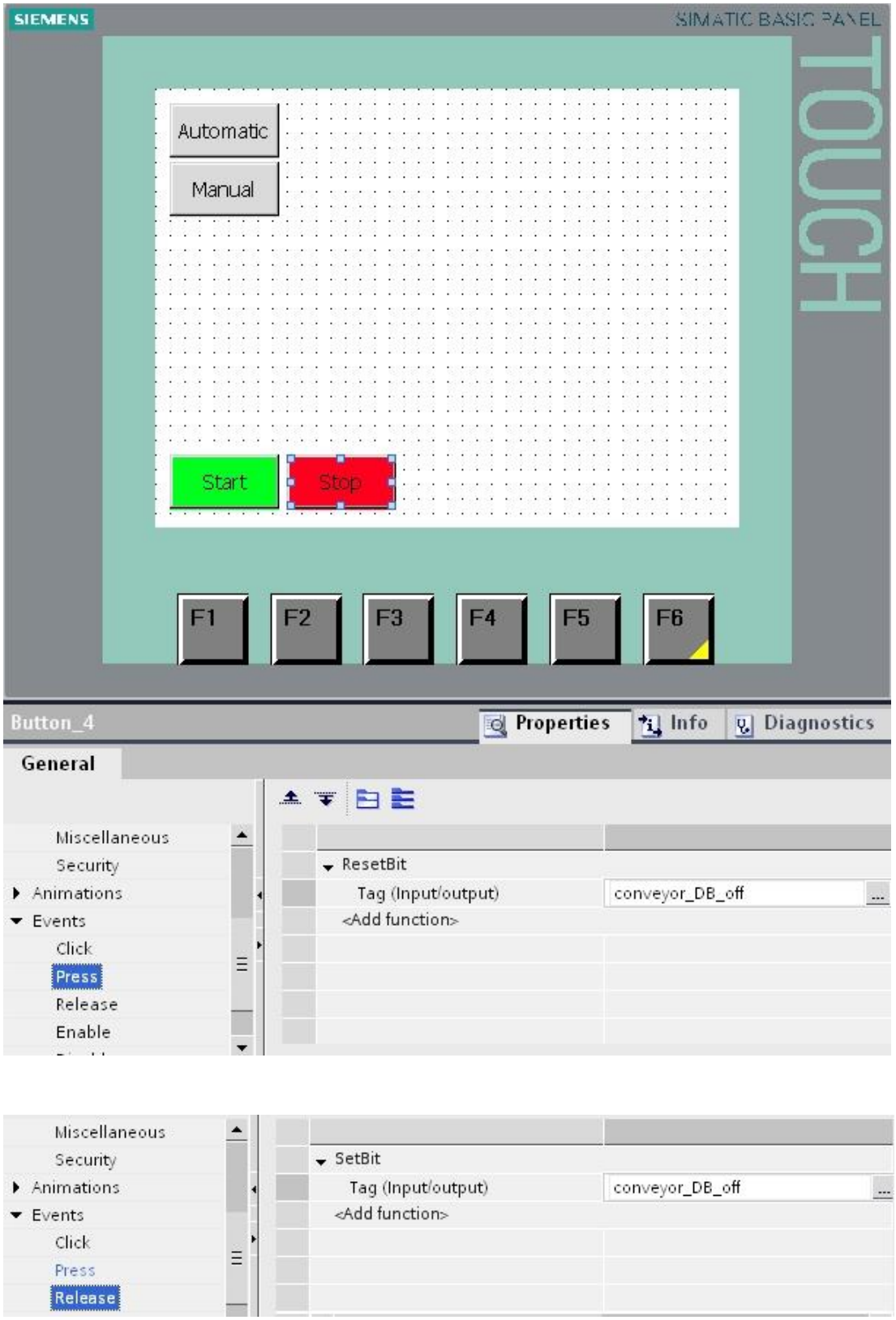
Next, we configure the start and stop buttons.
The Start button is created exactly like the automatic and manual buttons.
The Stop button has a break contact function and has to remove the signal when operated.

- Create the Start button
- Set the background color to Green
- Under Events Press, select under bit editing the function SetBitWhileKeyPressed.
- Then select the tag on in Conveyor_DB [DB1].



Next, do the following: Create the Stop button. Set the background color to Red.

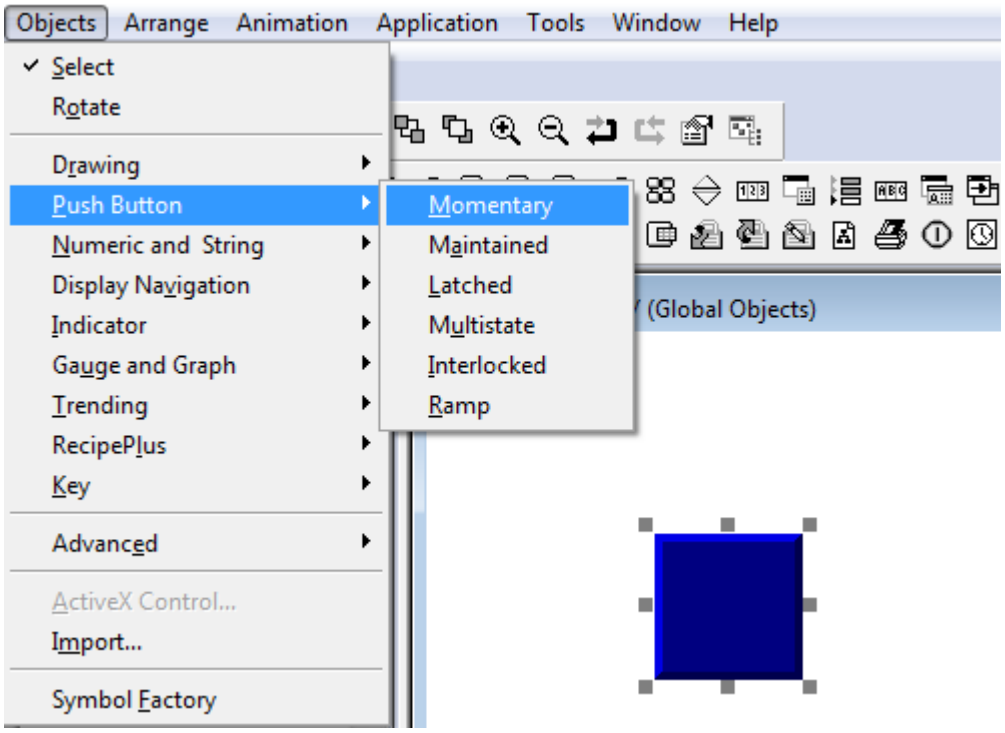
Under Events, at Press, select under Bit editing the function ResetBit and at Release the function SetBit with the tag off in Conveyor_DB [DB1].



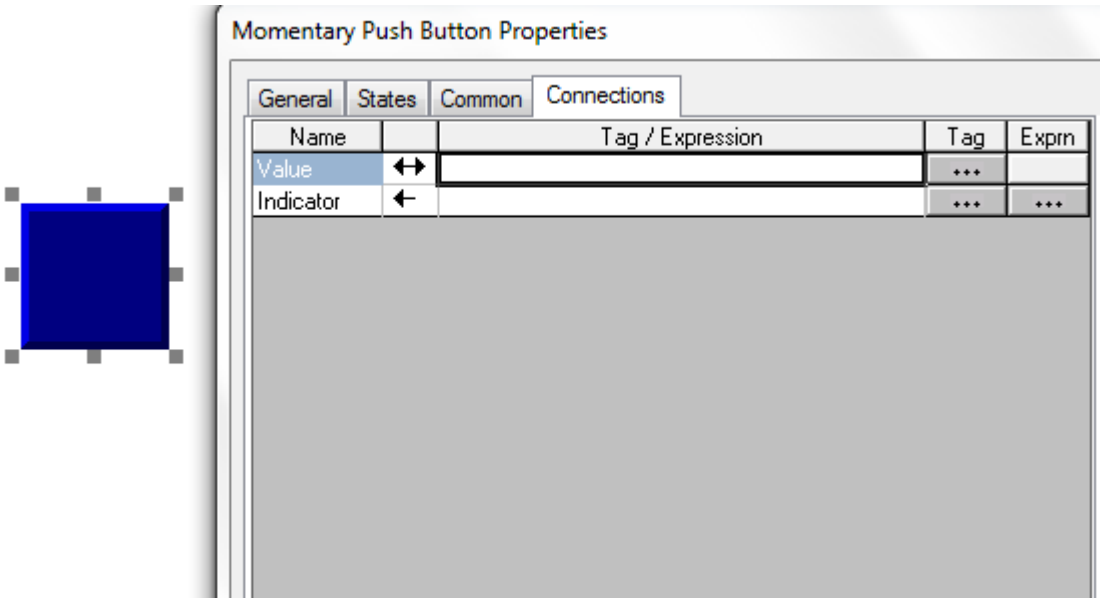
Screen with Start and Stop Buttons

Allen-Bradley Button Configuration

The choice of button type indicates the type of function desired. There is no need to program both press and reset but rather only type and the function is performed.



Choice of Button Type Determines Function



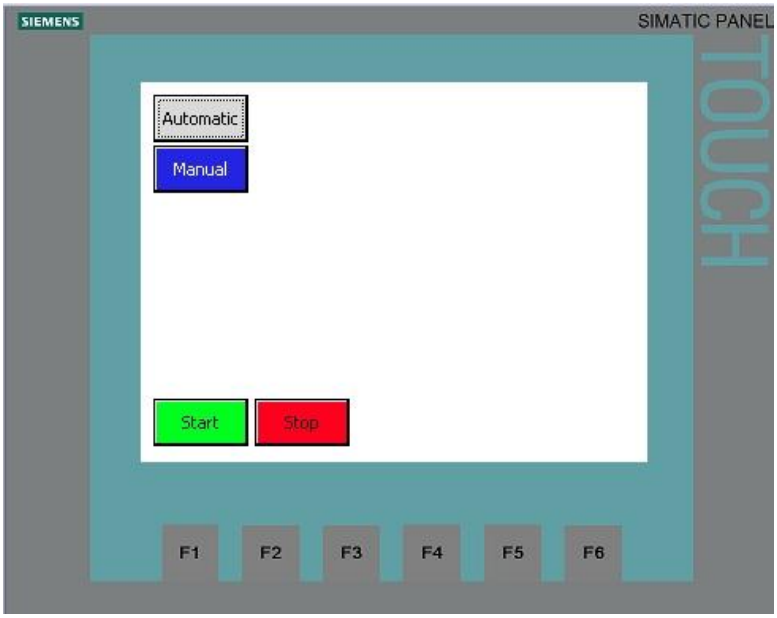
Connections Tab on Push Button Function

The tag or expression may be programmed for the value as well as for an indicator. The indicator actually is a second button with the state desiring to be displayed. This value may be the same address as the value entry or a second address. To perform a similar function, Siemens may require two buttons overlaying each other.

Before we test the visualization, first another change has to be made in the Step7 program In OB1, remove the assignment S3 and S4 when calling FB1.

Save and load the modified program.

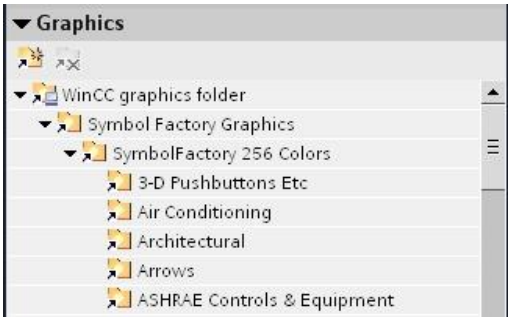
Load the configuration to the panel, and test the Start and Stop buttons.

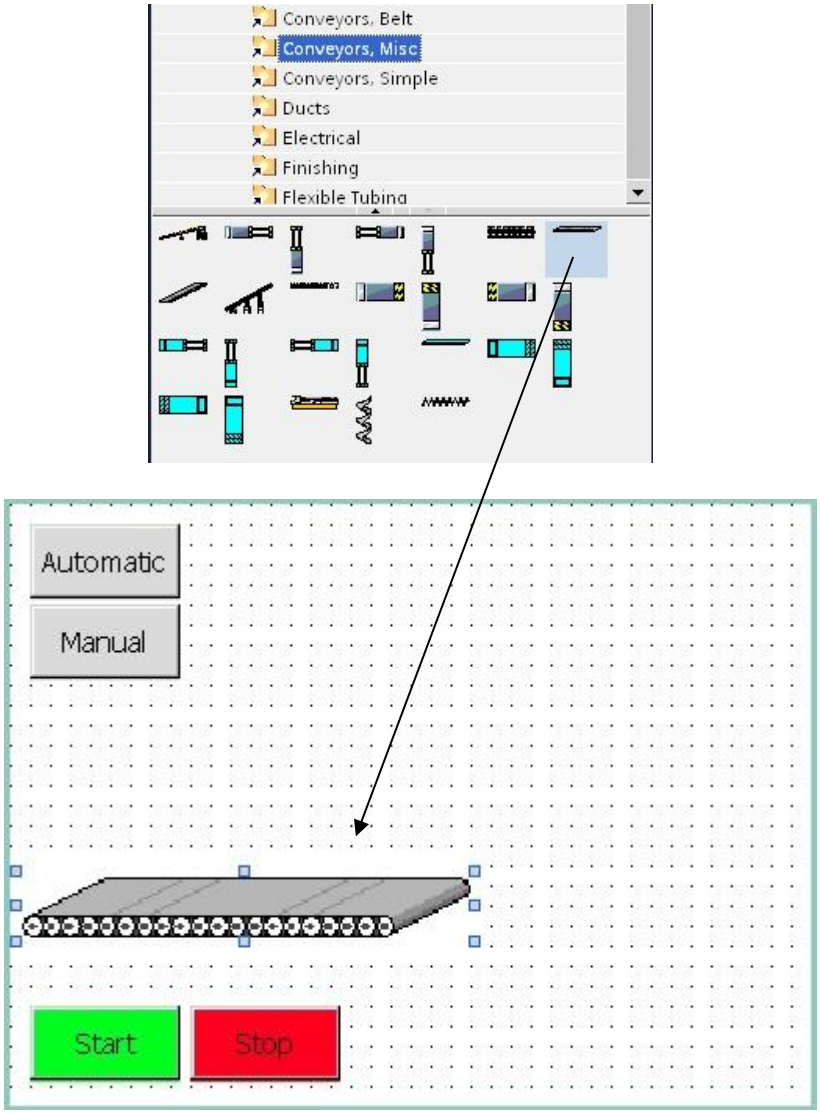


Inserting Graphics from the Graphics Folder

In the tool box under Graphics, open the directory tree WinCC graphics folder\SymbolFactory 256Colors\Conveyors, Misc.

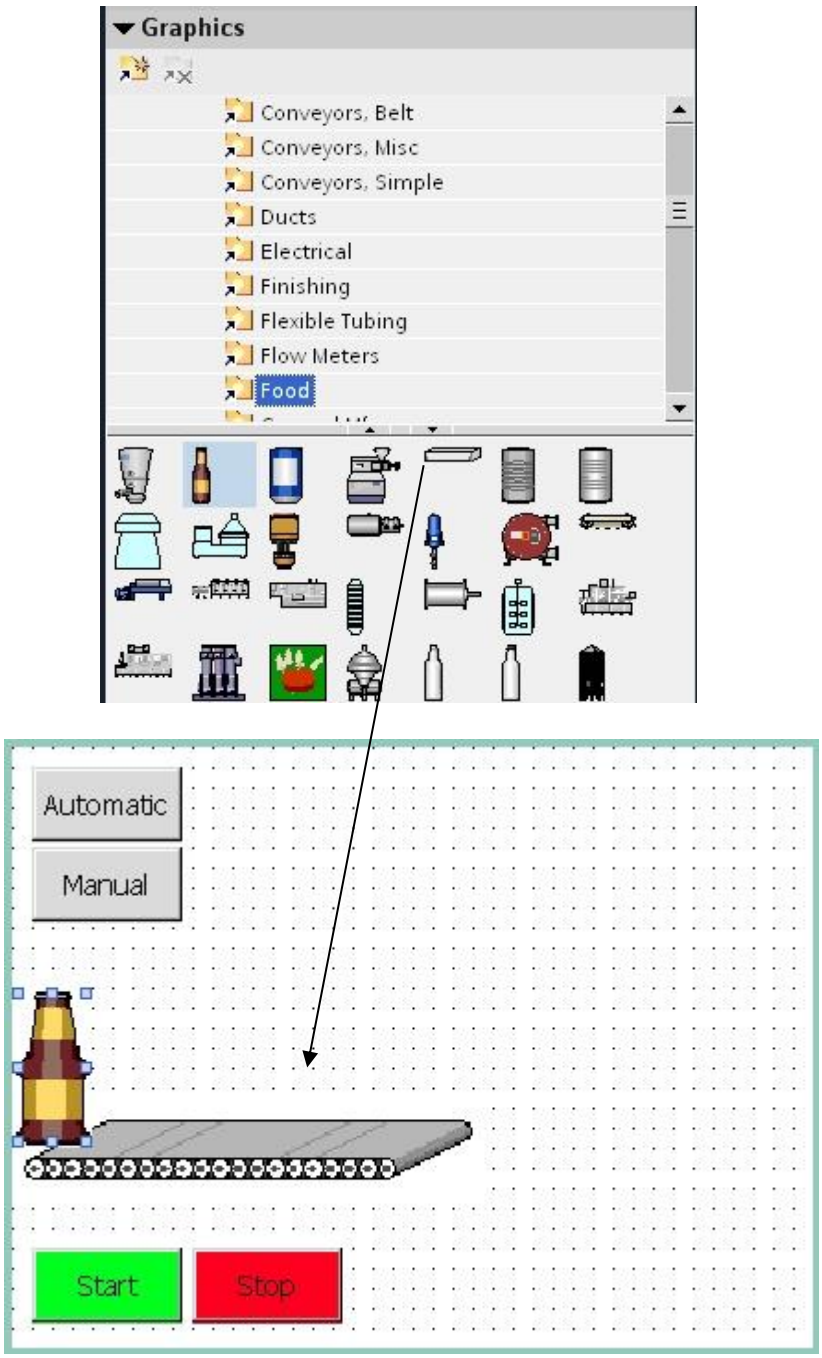
Drag and drop the graphic of the conveyor belt to the root screen.





In the tool box under Graphics, open the directory tree WinCC Graphic folder\SymbolFactory 256 Colors\Food.

Then drag and drop the picture of the beer bottle in the root screen.
Change the size and the position of the bottle.



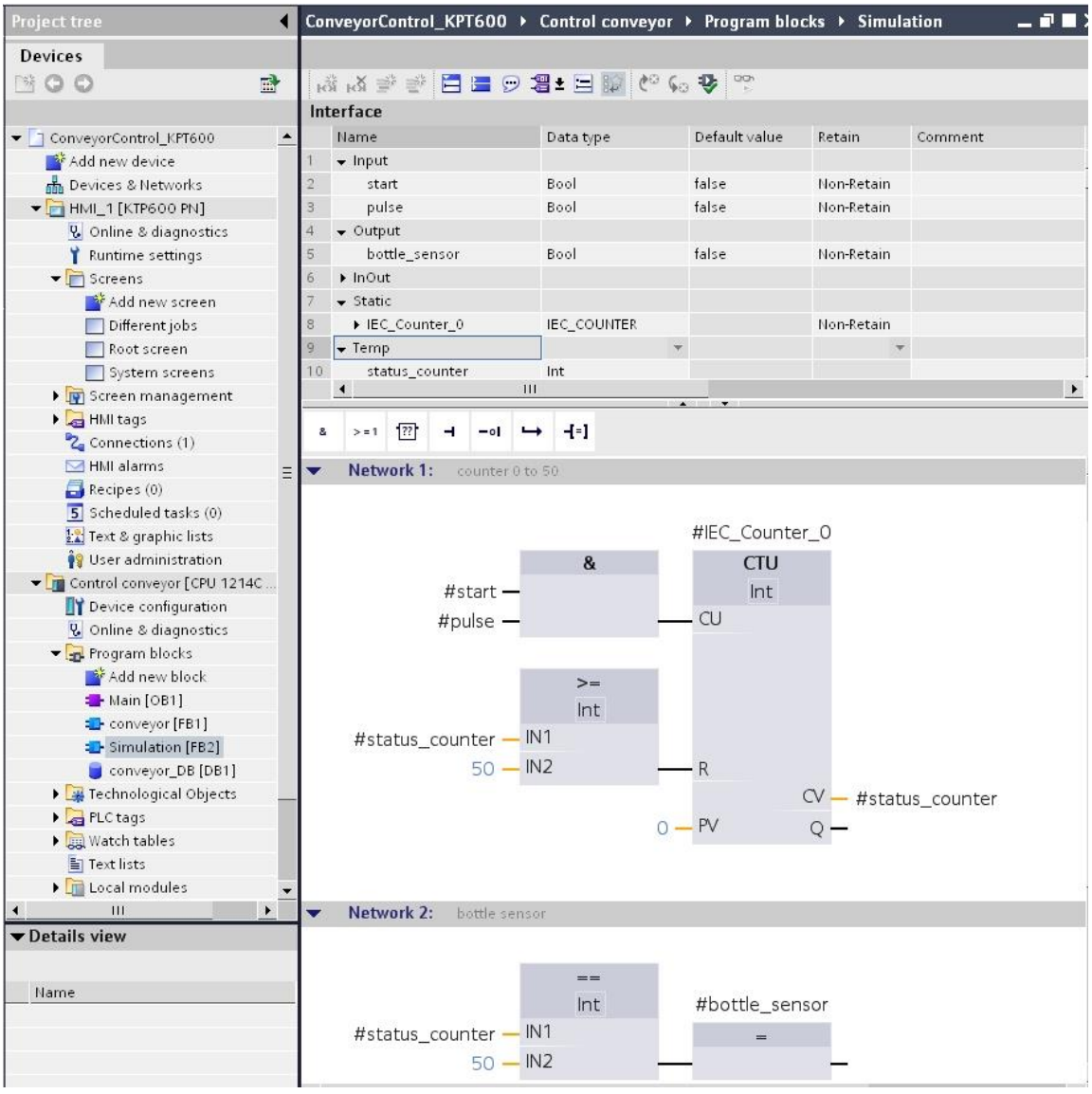
Note

All screen objects have to be within the work area (320x240 pixels).

Control Program for Simulating the Bottle Movement

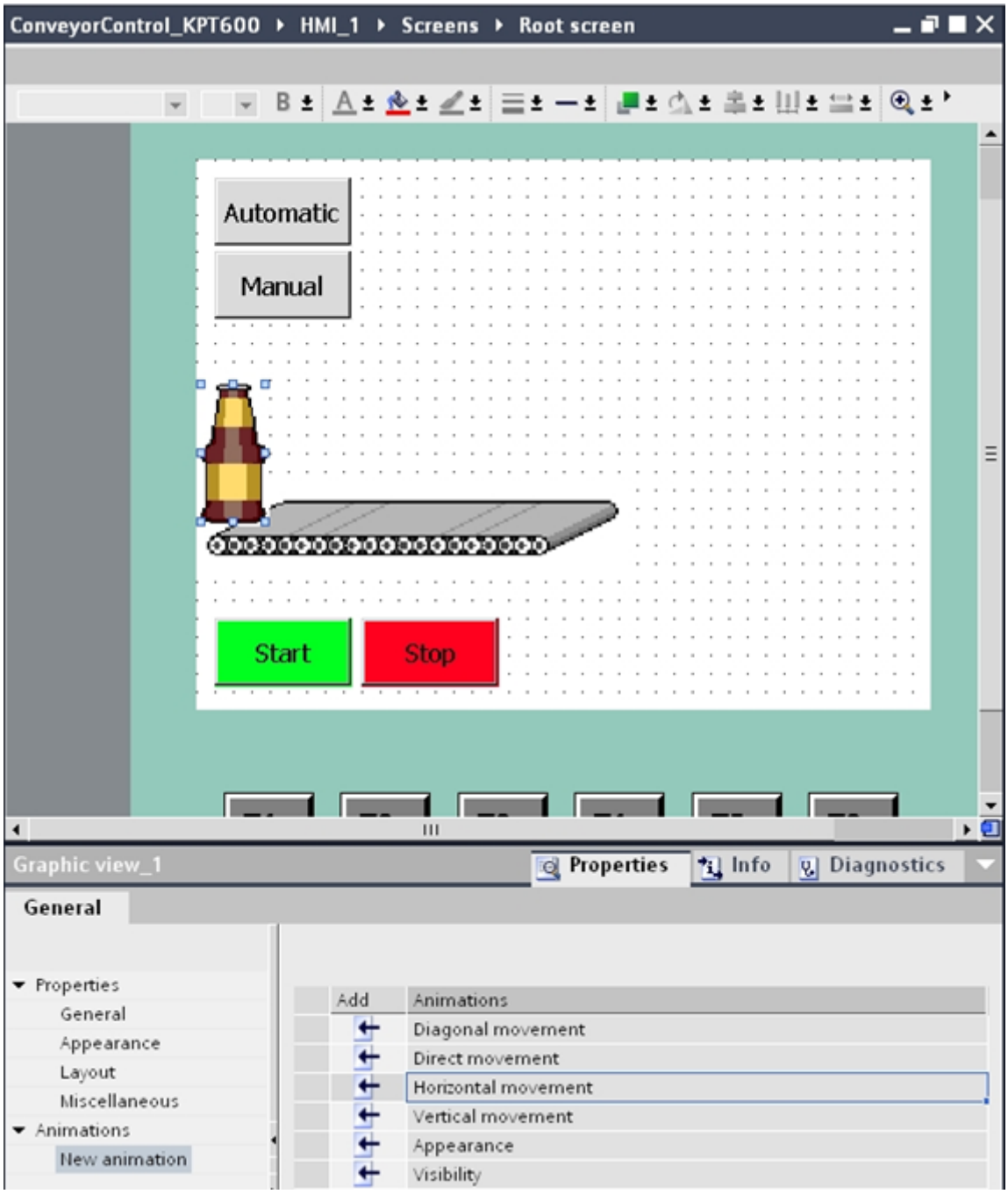
To simulate the bottle movement and the bottle sensor, we create a new block. The FB2 (simulation) below with tag declaration and networks consists of a counter that, through a start signal, always counts up from 0 to 50.

In Network 1, the CTU (count upward) is inserted as multi-instance.
In Network 2, a bottle sensor pulse signal is read out when the count 50 is reached.
This simulates when a bottle leaves the conveyor.



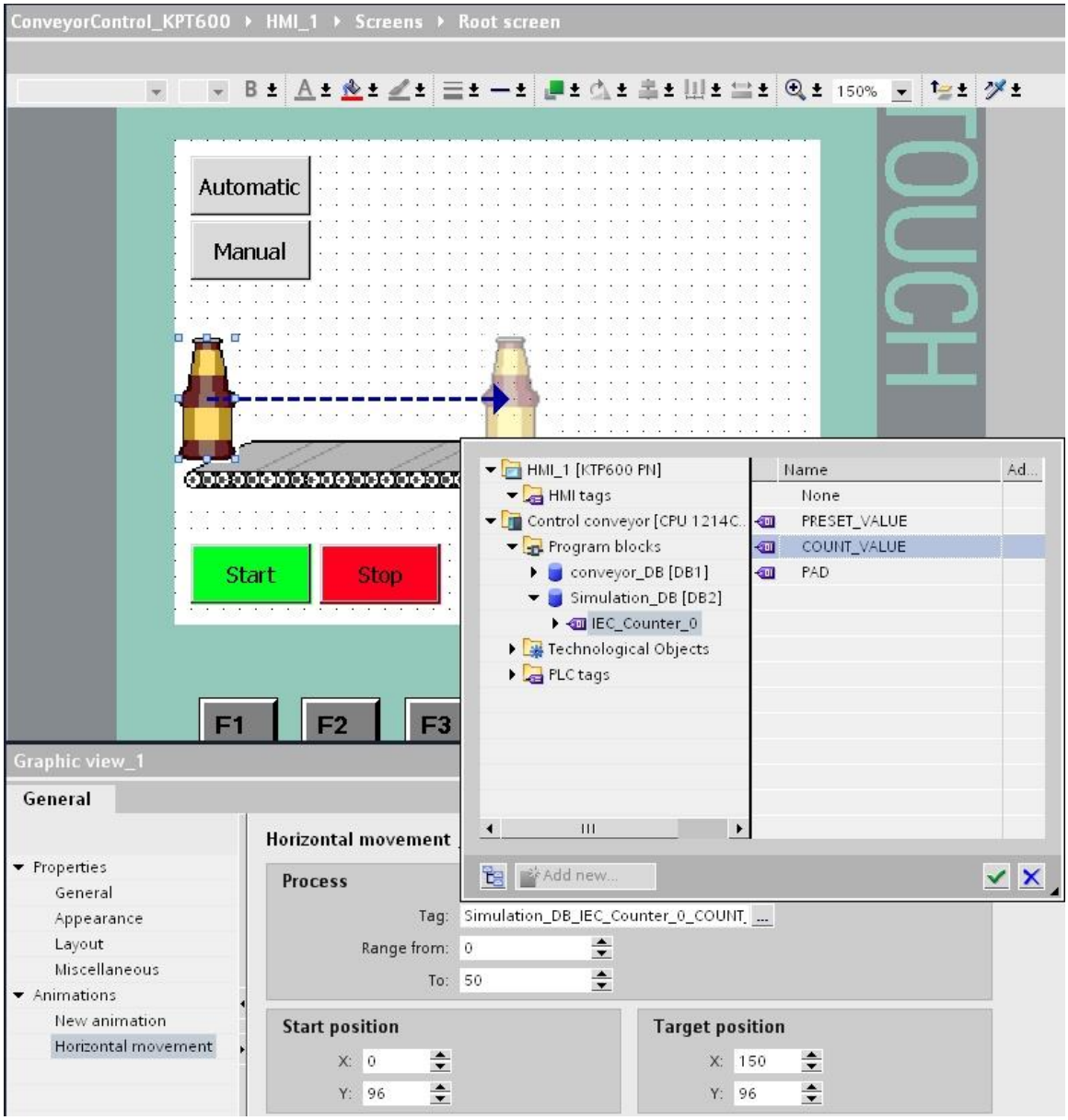
Configuring the Bottle Movement

Highlight the bottle and select under the tab Properties at Animations - Horizontal movement (double click).



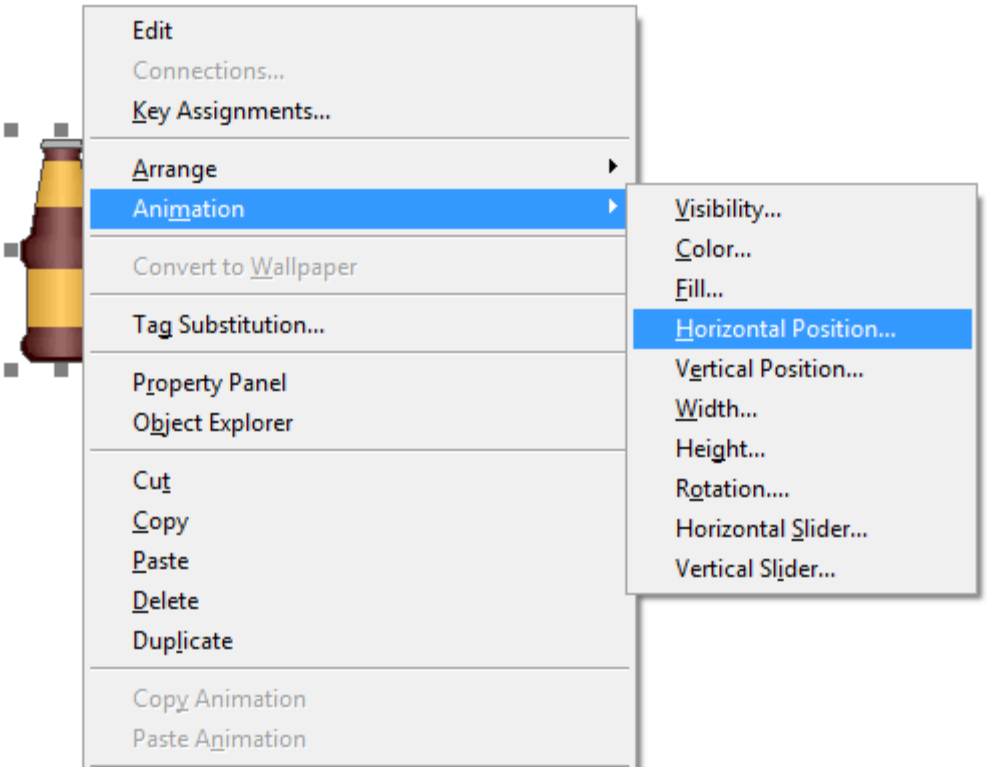
As variable, select COUNT_VALUE of the Simulation_DB (DB2).
For range, enter from 0 to 50.

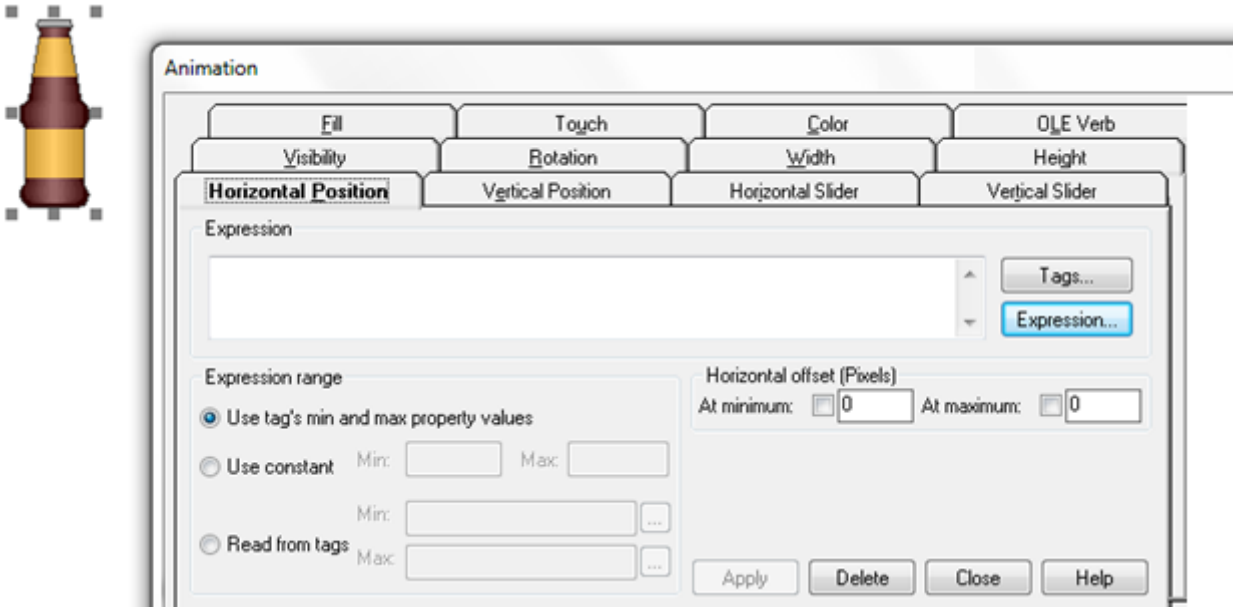
Change the bottle's target position up to the conveyor end X150.



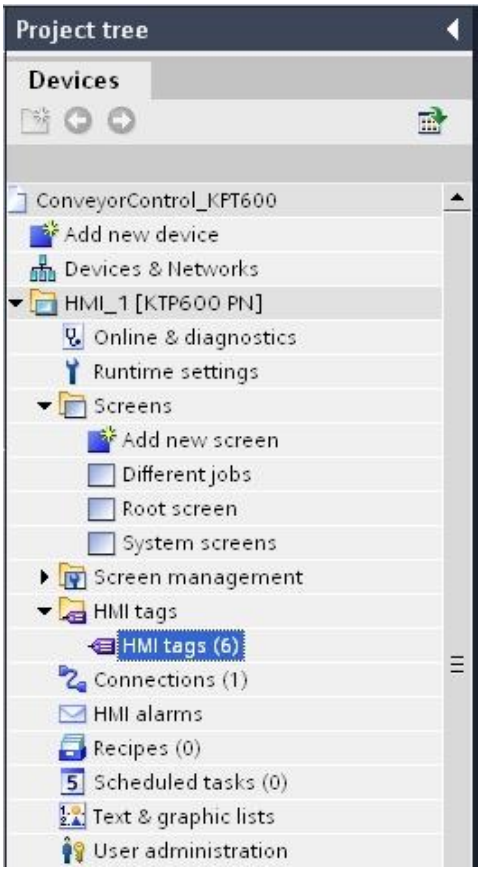
Allen-Bradley Animation

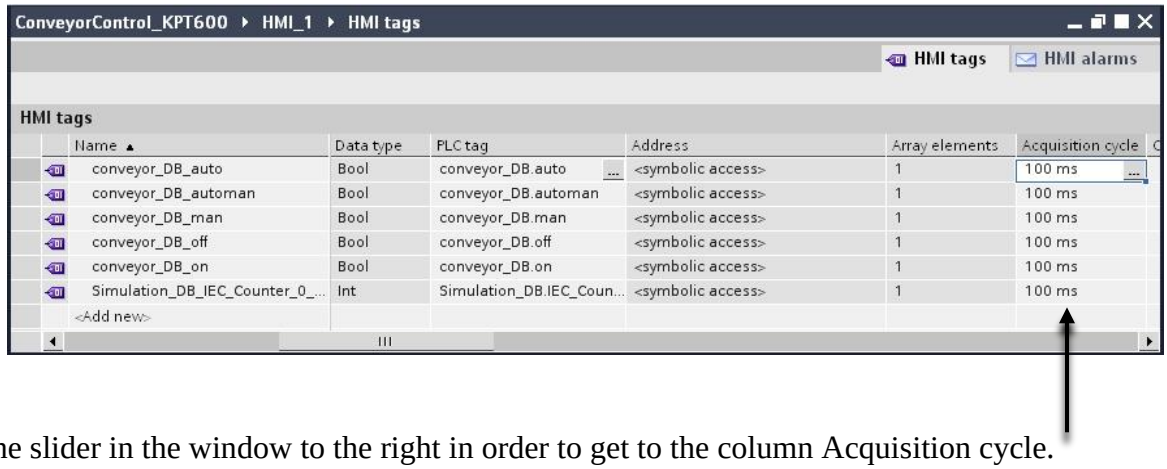
The Allen-Bradley animation proceeds much the same as Siemens in that the device edited may be animated in a number of ways including horizontal position. The position is a function of a number in a location which is monitored and the beer bottle moved accordingly.





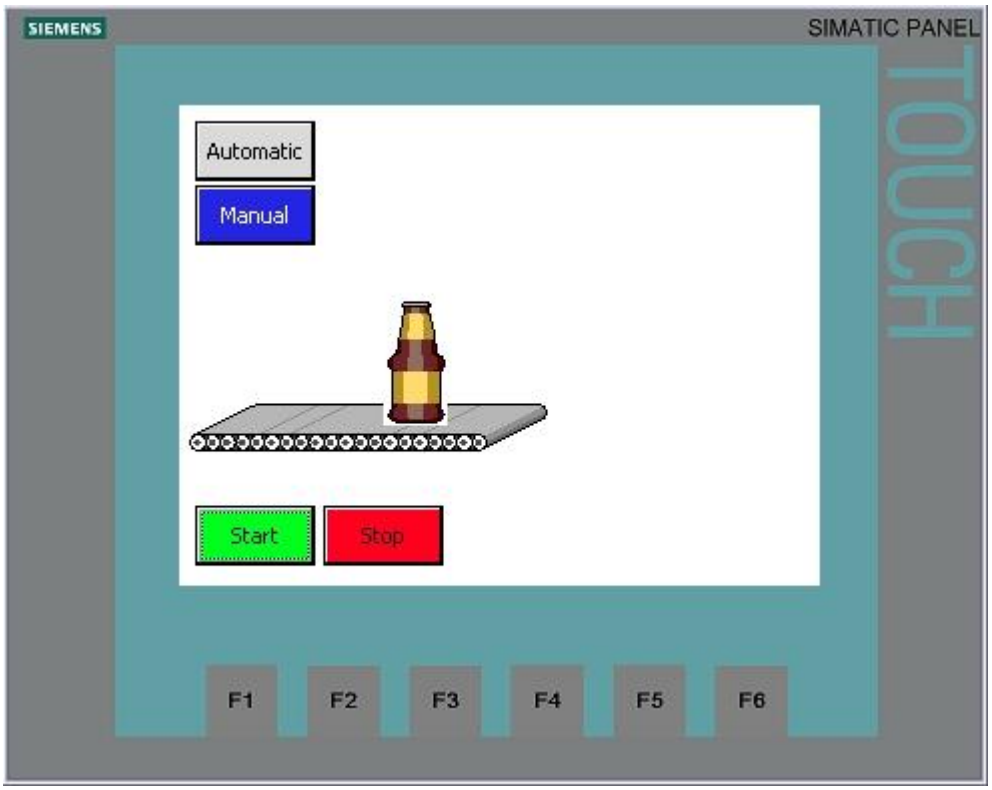
In the project window, select the HMI tags.

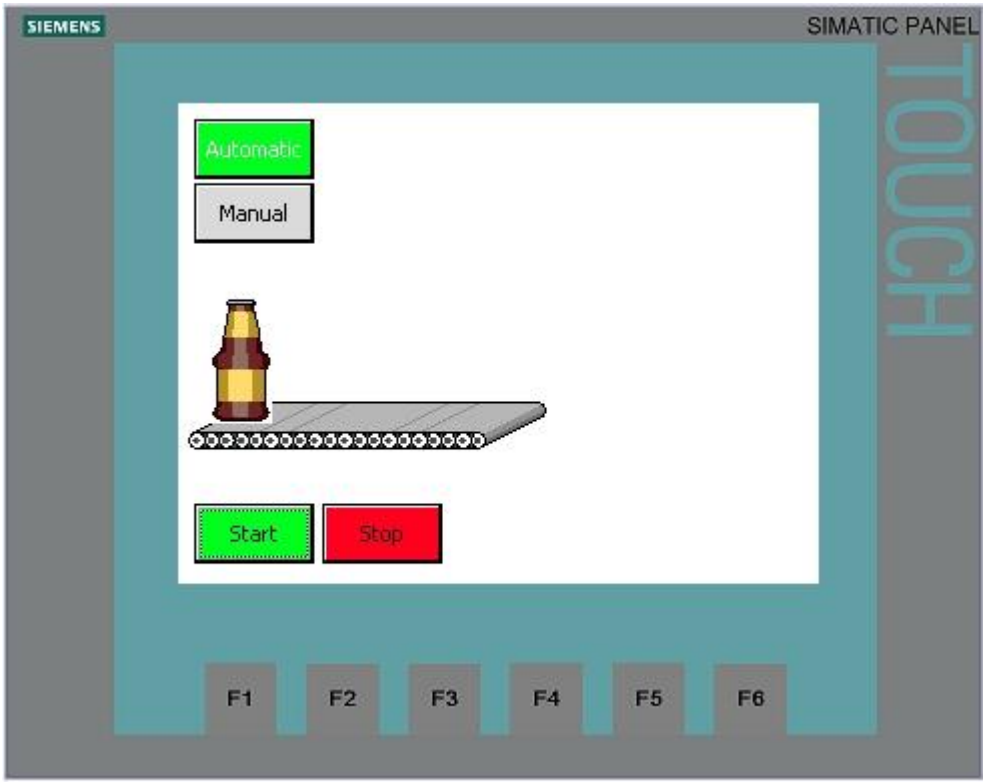




Drag the slider in the window to the right in order to get to the column Acquisition cycle.
Set the acquisition cycle of the HMI tag to 100ms.

Then save the project, load it to the panel and test it.

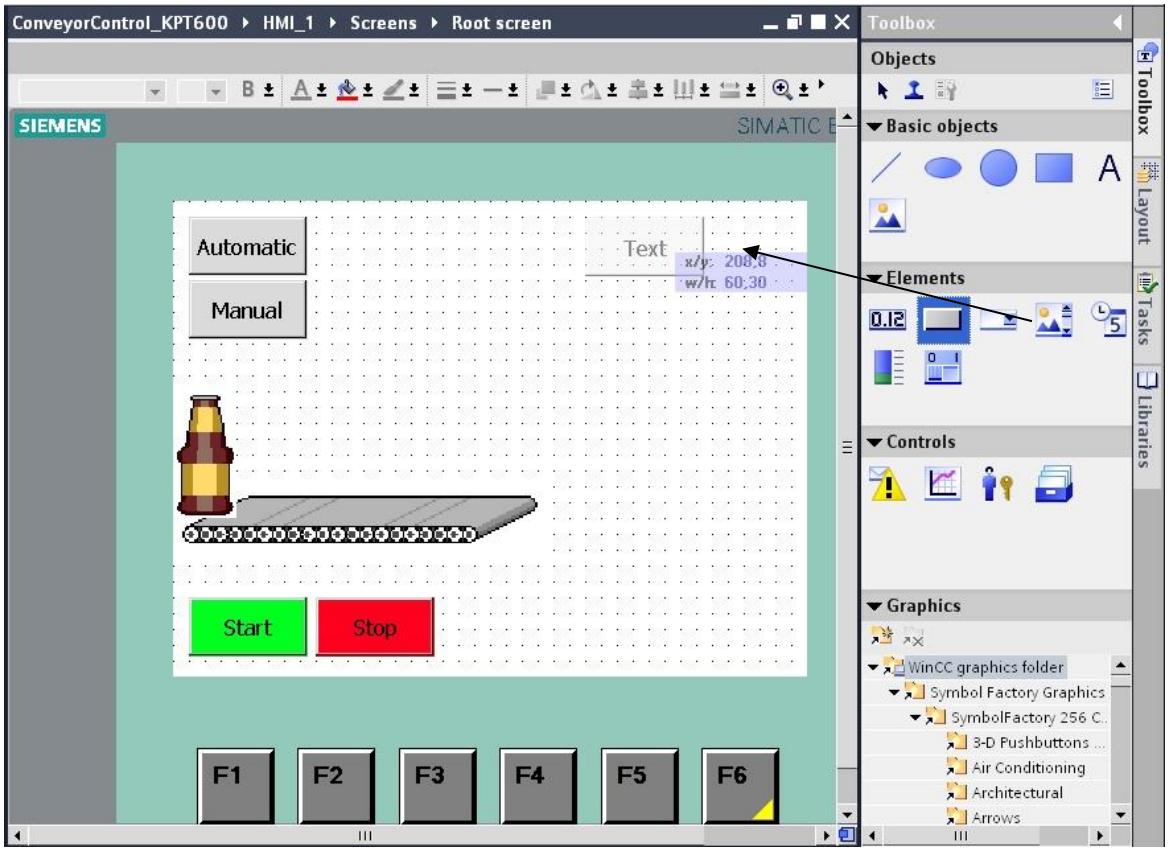




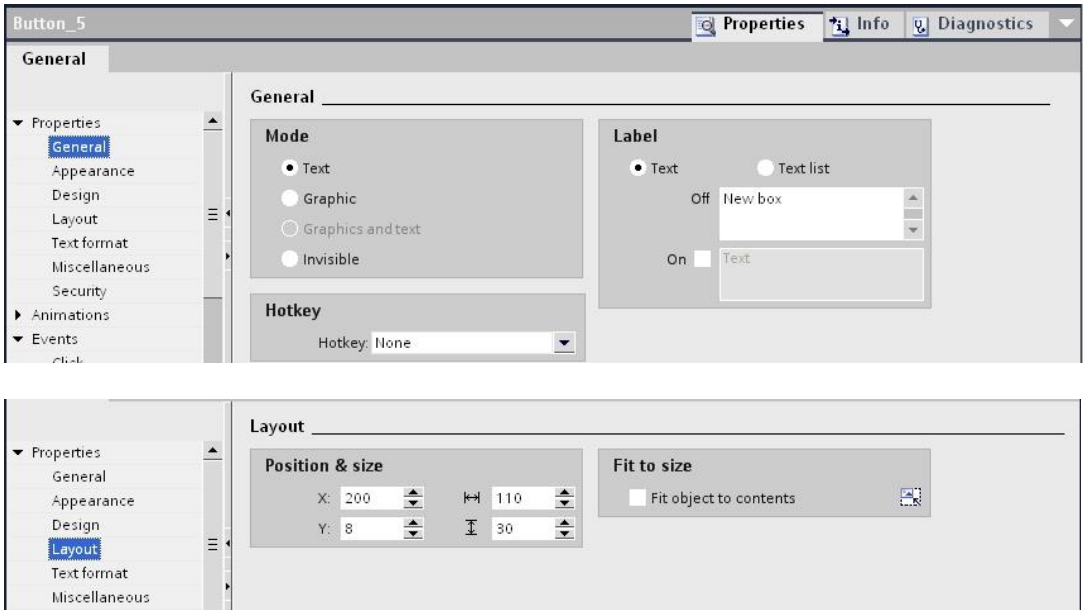
After 20 bottles, the conveyor motor stops. The bottle counter has to be reset before the next start.

Resetting the Bottle Counter

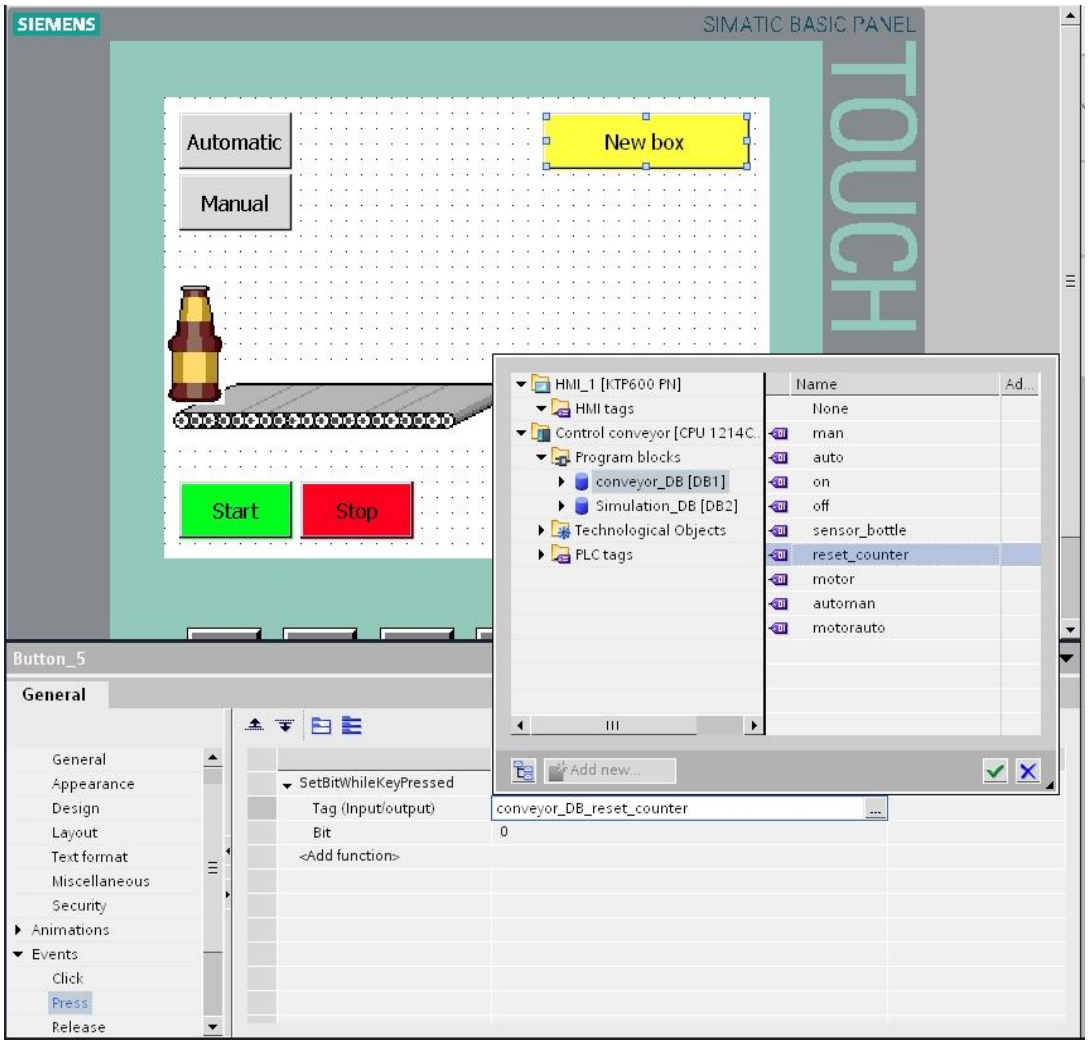
Drag a button into the root screen.



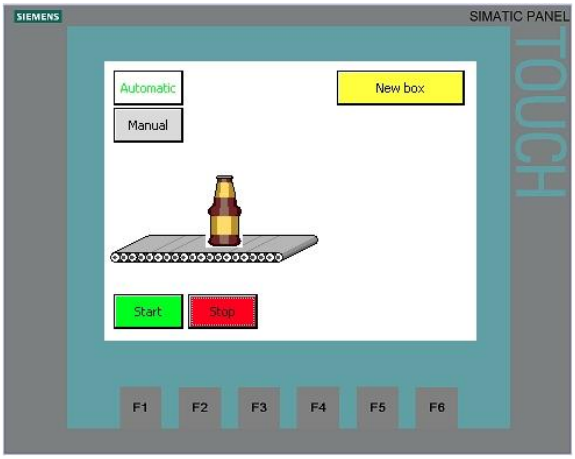
As text, enter Exchange beer case and adjust the color Position & size to the button.



Under Events Press, select under bit editing the function SetBitWhileKeyPressed.
Select the tag reset_counter from Conveyor_DB [DB1].

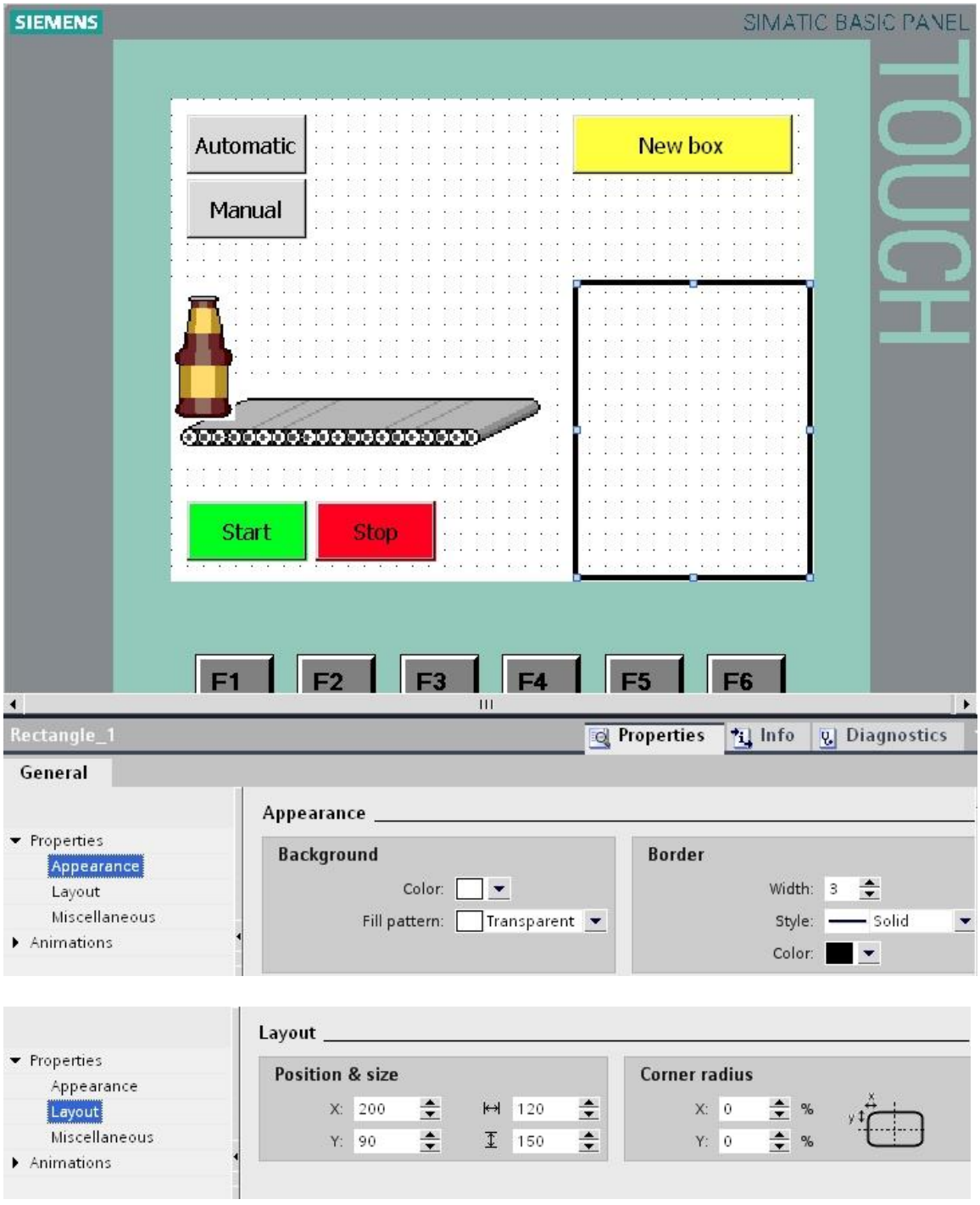


Set the acquisition cycle of the new HMI tag to 100ms.
Then save the project, load it to the panel and test it.

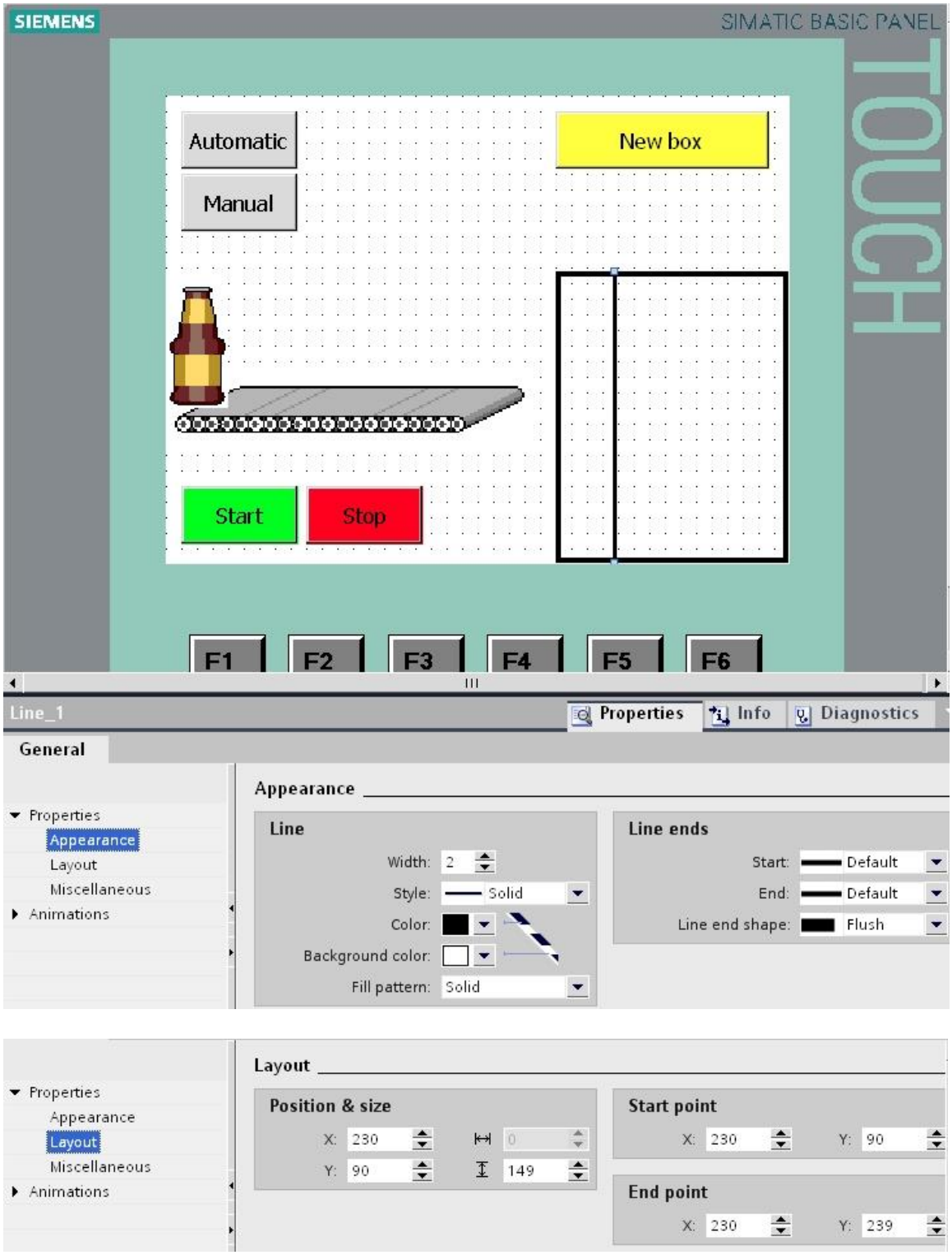


Drawing the Beer Case

Draw a rectangle with a transparent background.
Enter the width of the border, the position and size.



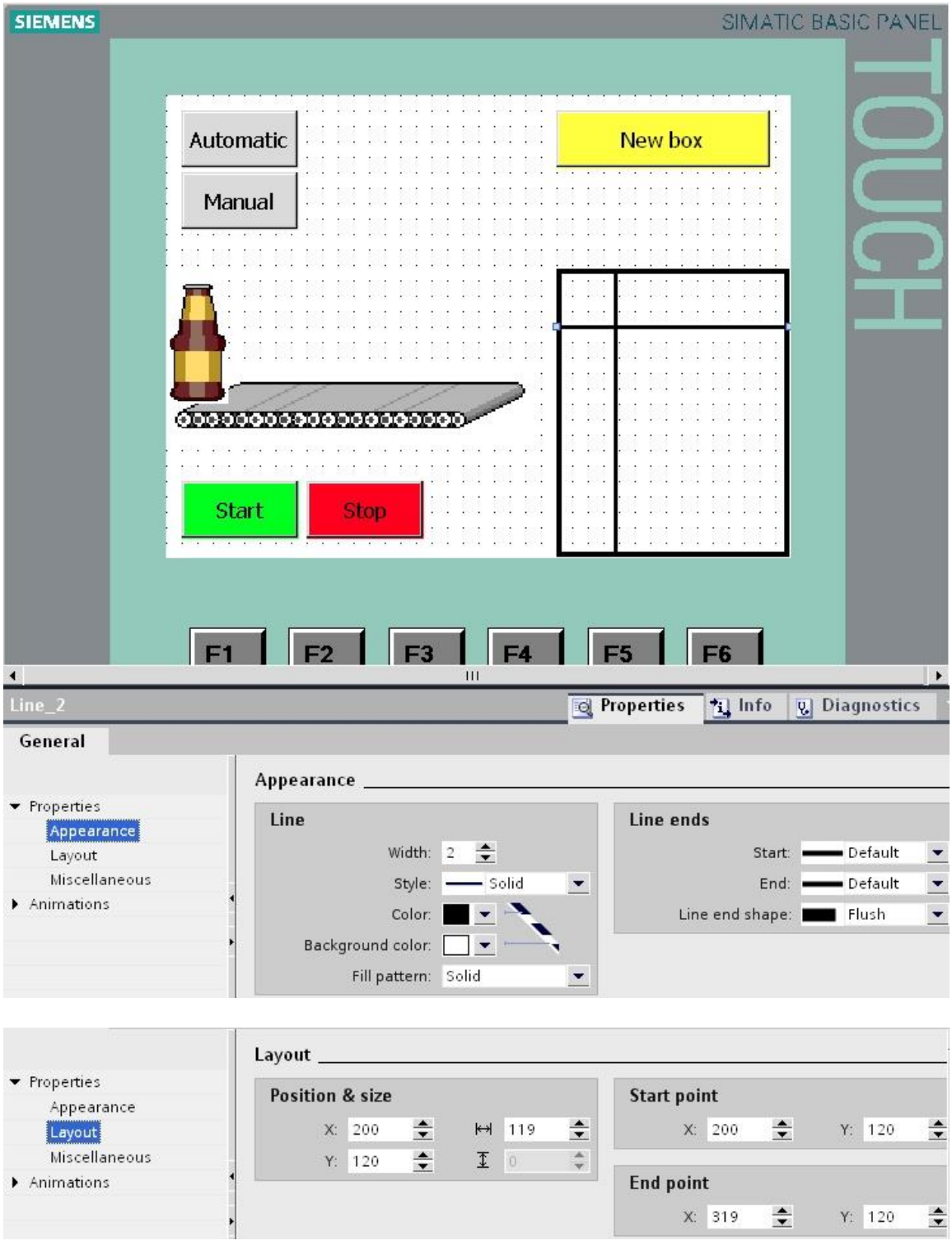
Draw a vertical line at a distance of 30 pixels.



Note

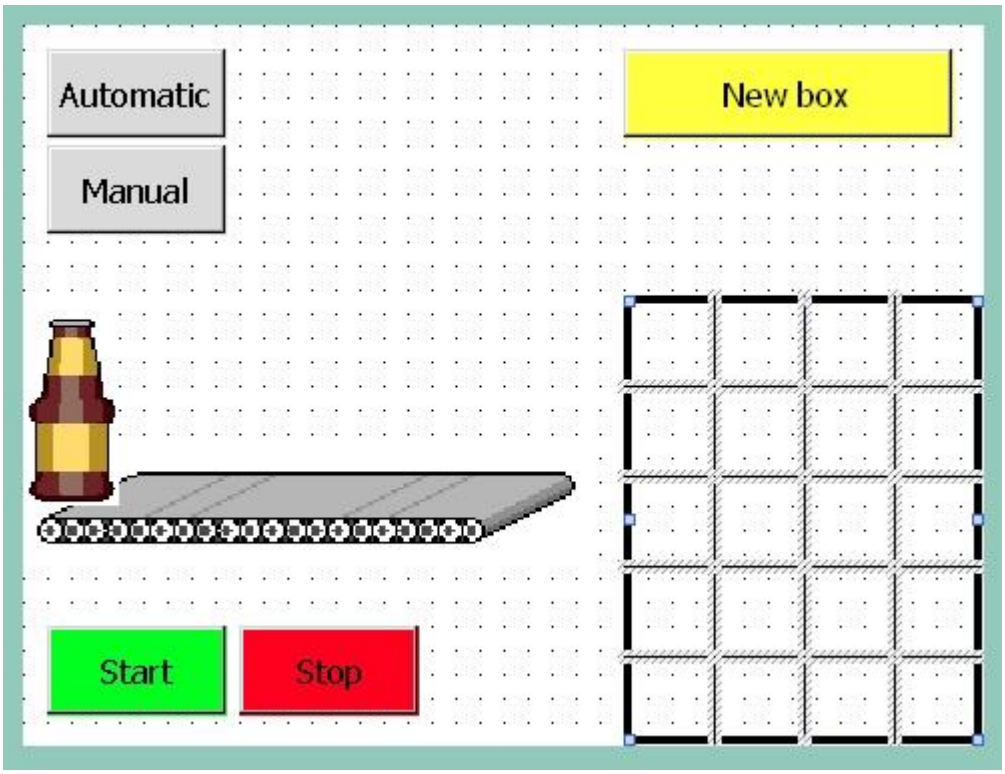
Although the measurements of the line are correct, it is drawn beyond the rectangle.
Change the form of the line ends to flush, and shorten the line by one pixel from 150 to 149.

Draw a horizontal line spaced at 30 pixels

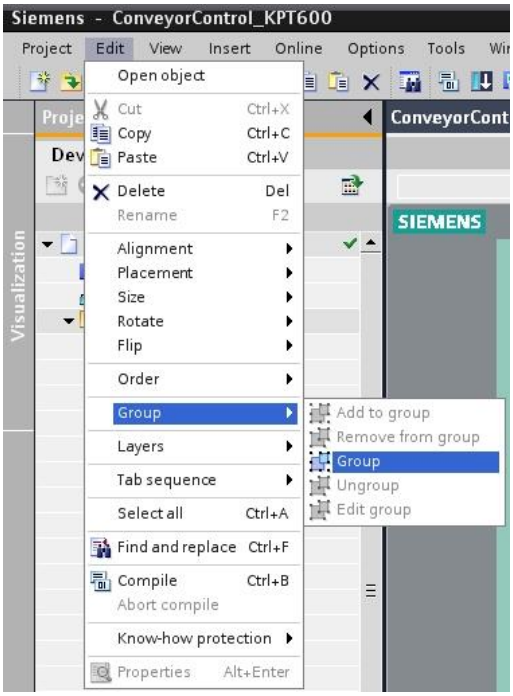


With copy and paste, create the remaining lines spaced at 30 pixels.

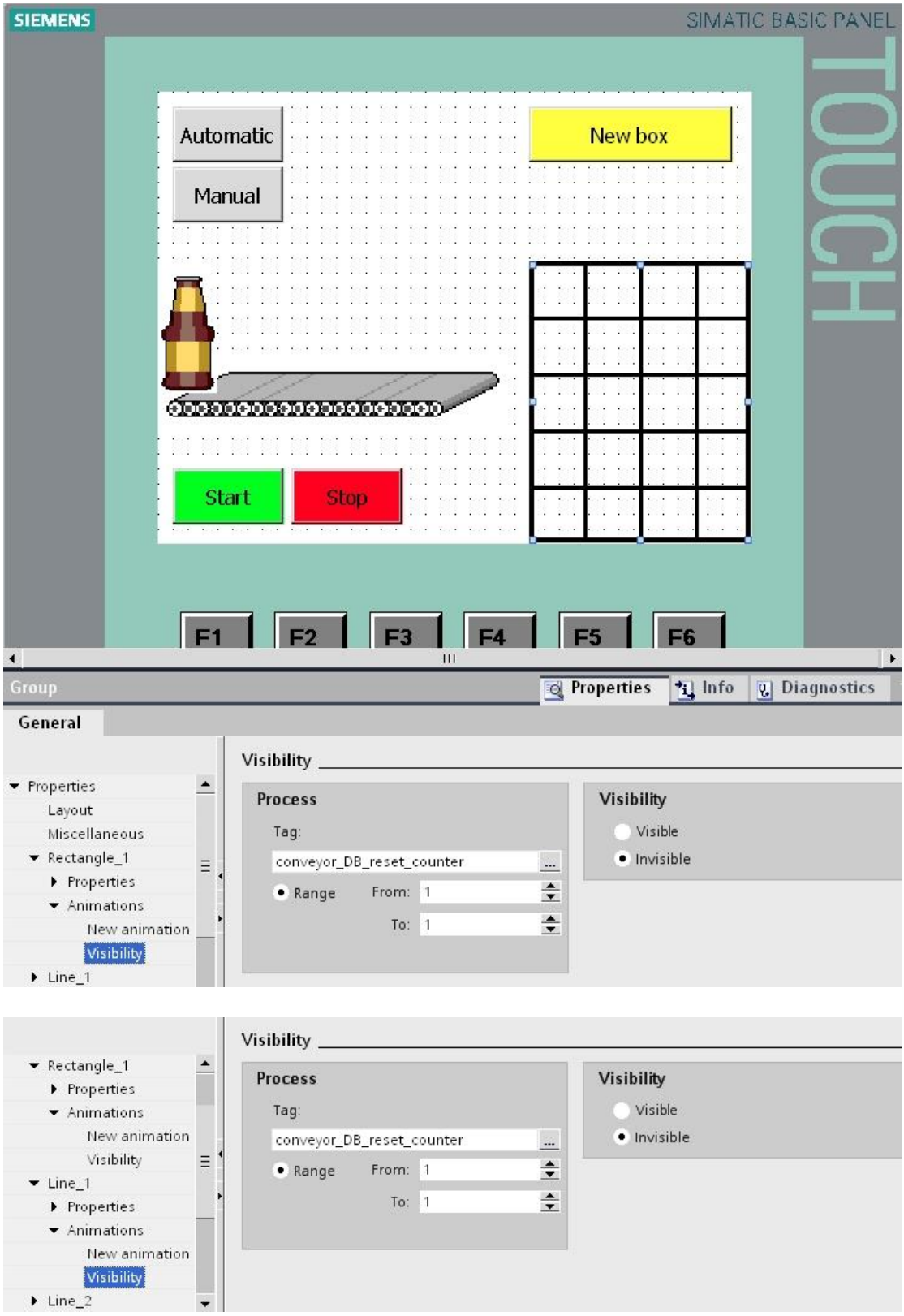
Highlight the beer case by dragging a border around it with the mouse.



In the menu Edit select the function Group



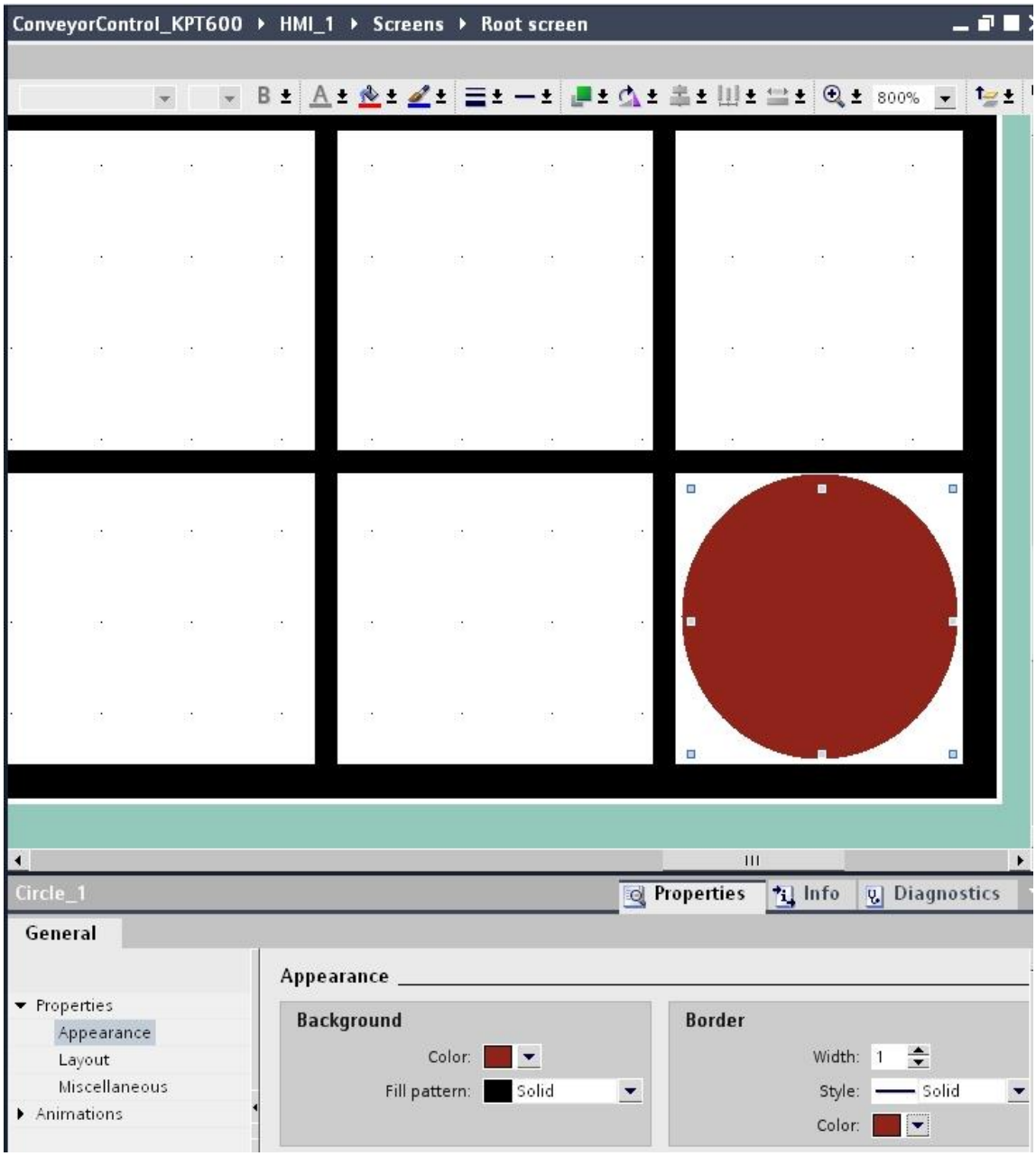
We don’t want to display the rectangle and the lines when the beer case is exchanged. At Rectangle_1 and at the lines, generate the animation Visibility using the tag Conveyor_DB_reset_counter at value 1 Invisible. For the lines, the animation can also be copy/pasted.

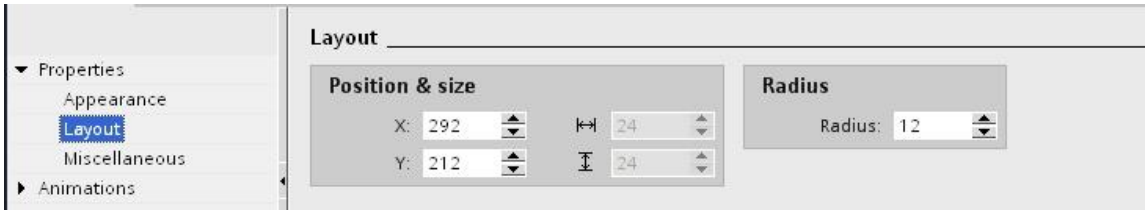


Then save the project, load it to the panel and test it.

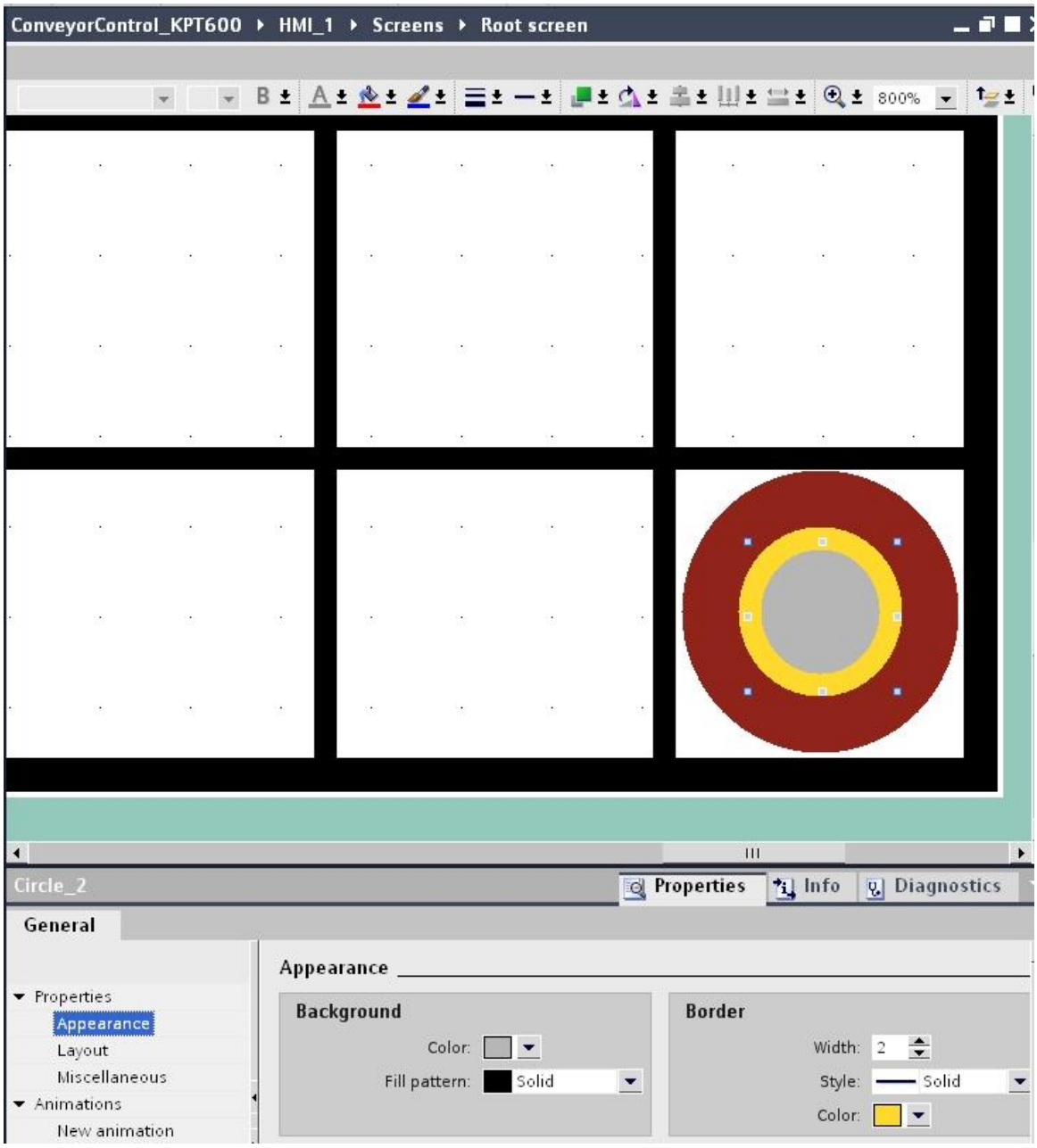
Drawing Bottles in the Case

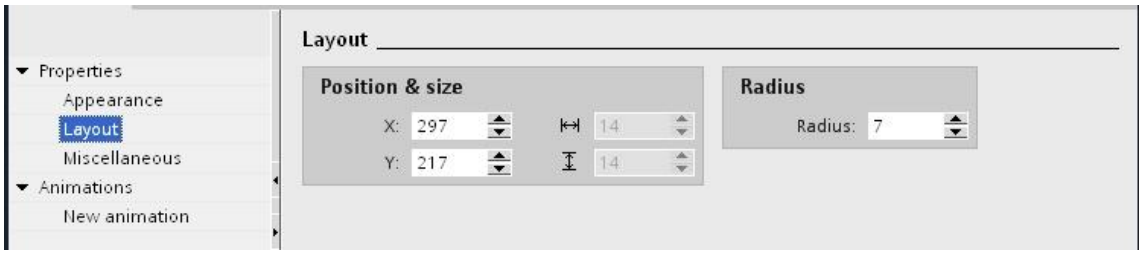
Enlarge the view and draw a circle in the lower right field of the box.





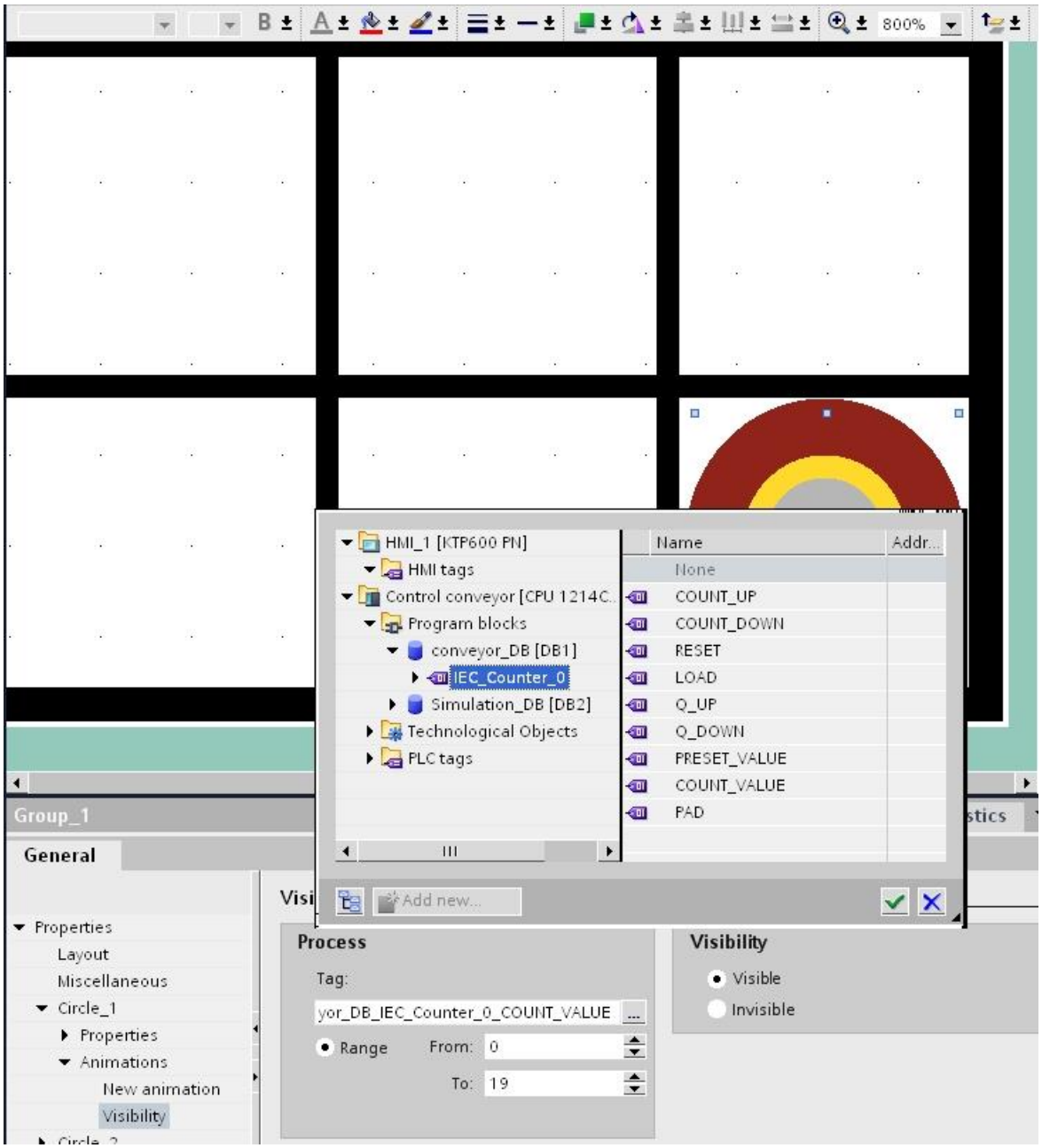
Draw a second circle.

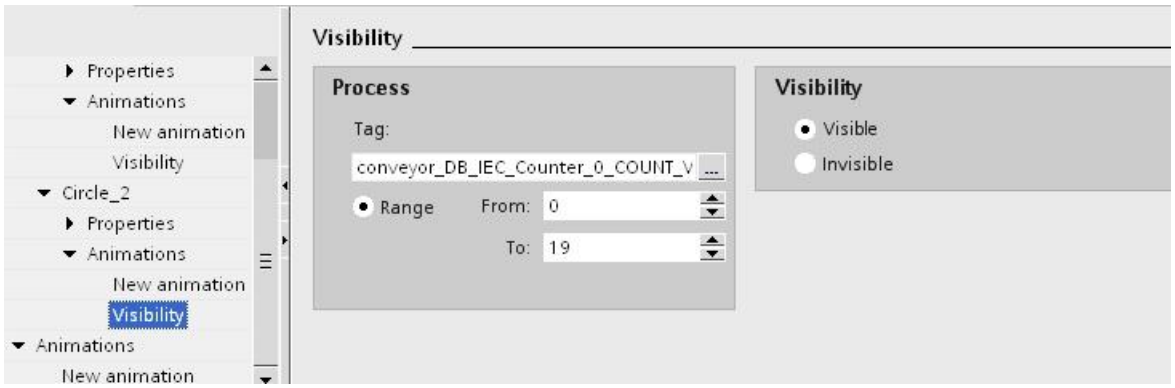




Group the two inserted cycles.

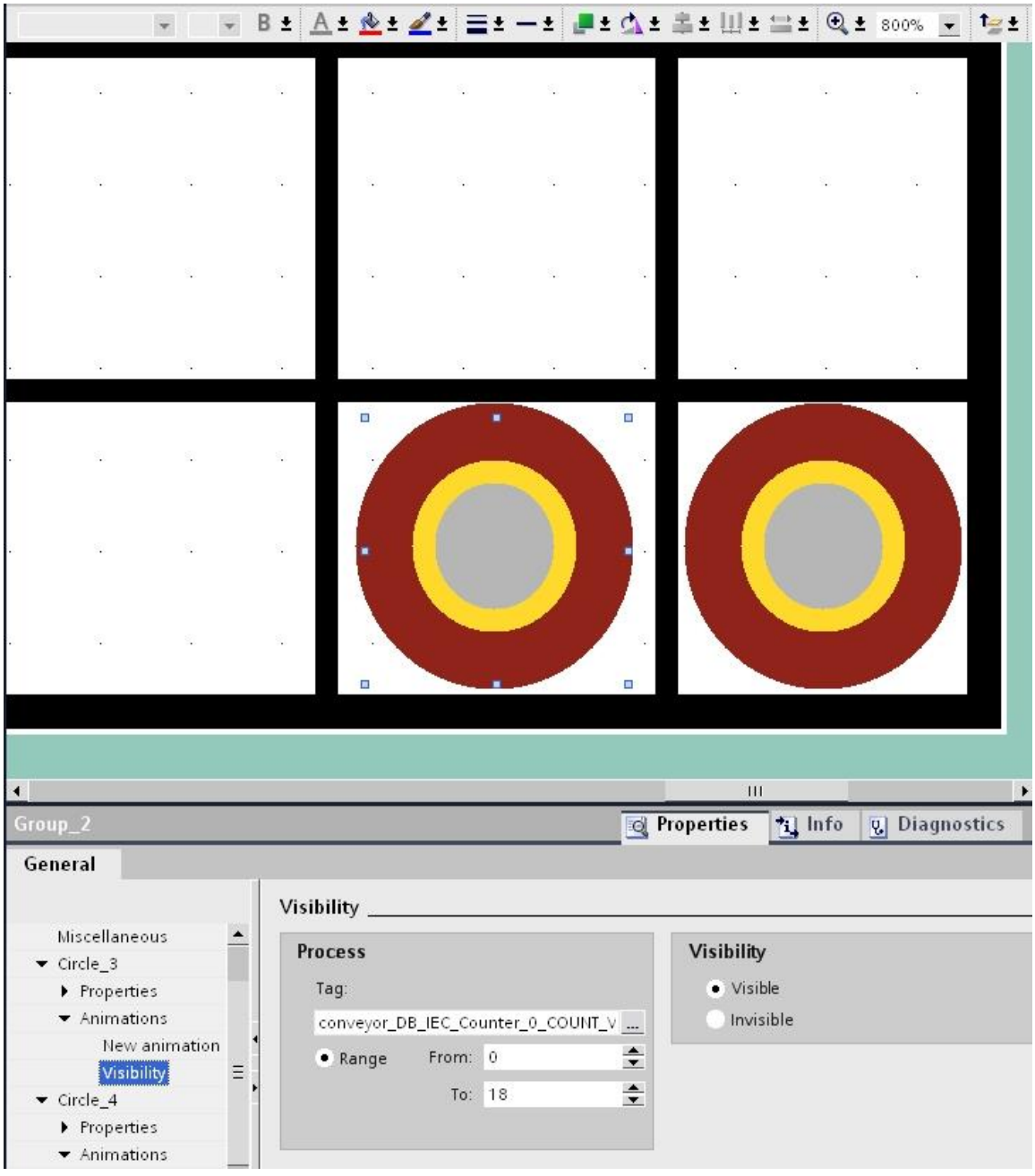
At Circle_1 and Circle_2, generate the animation Visibility with the tag Conveyor_DB_IEC_Counter_0_COUNT_VALUE value range 0 to19 Visible.

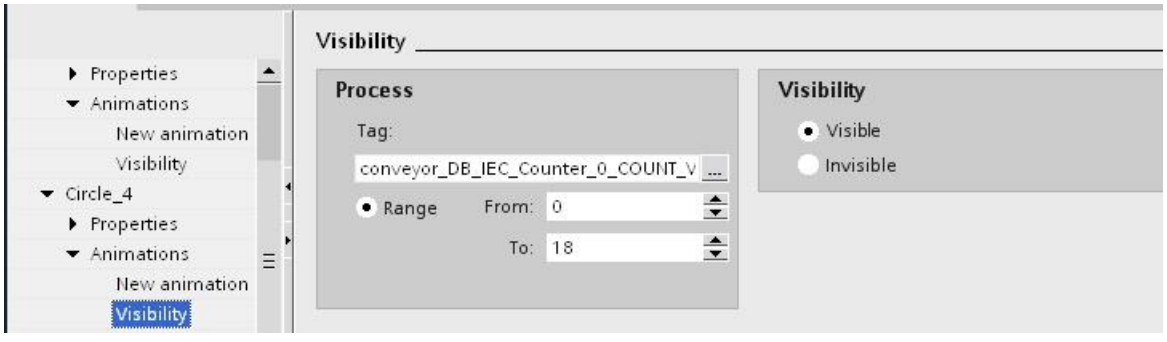




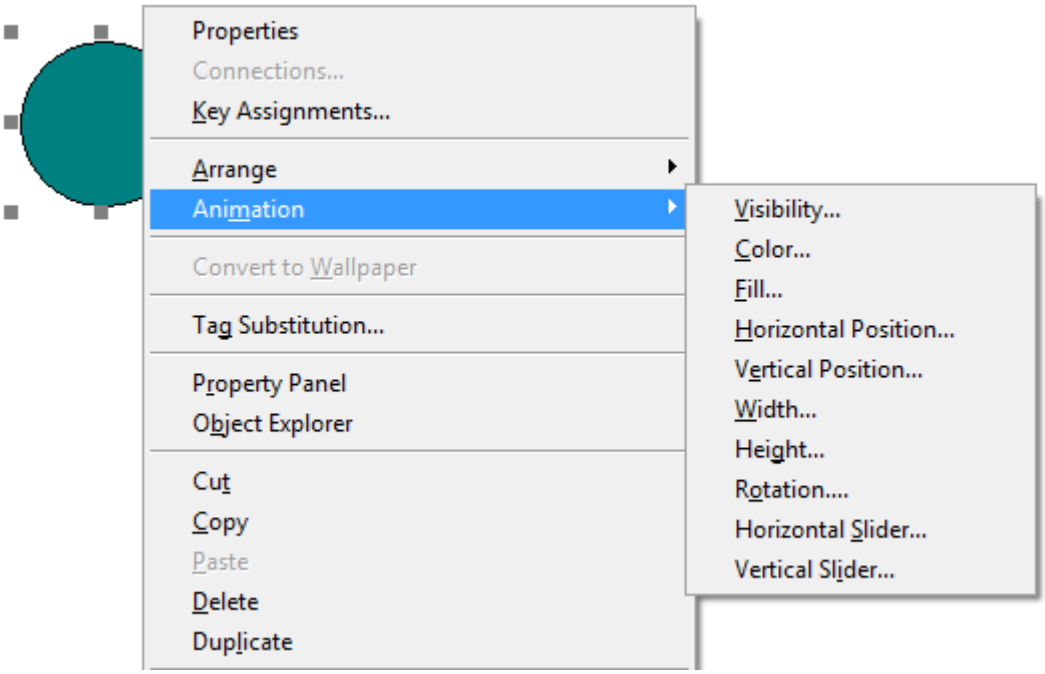
Copy and paste the bottle.

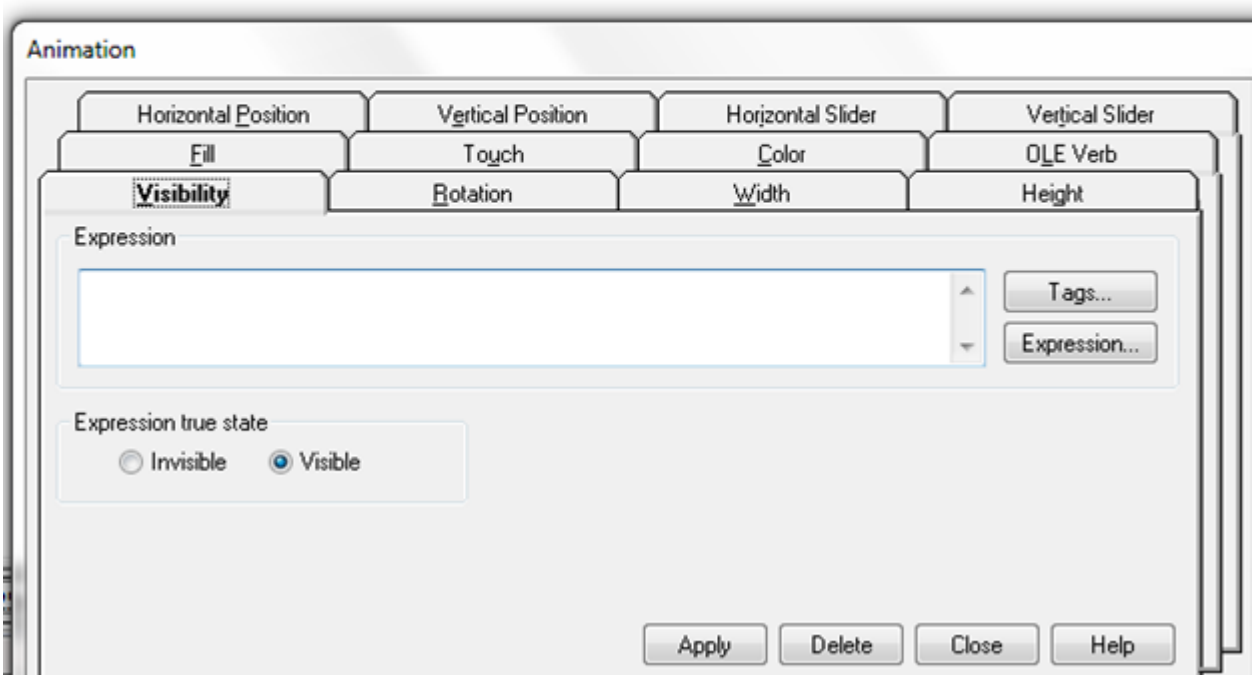
For both circles, under Visibility change the value range of the tag Conveyor_DB_IEC_Counter_0_COUNT_VALUE to 0 to18 Visible.



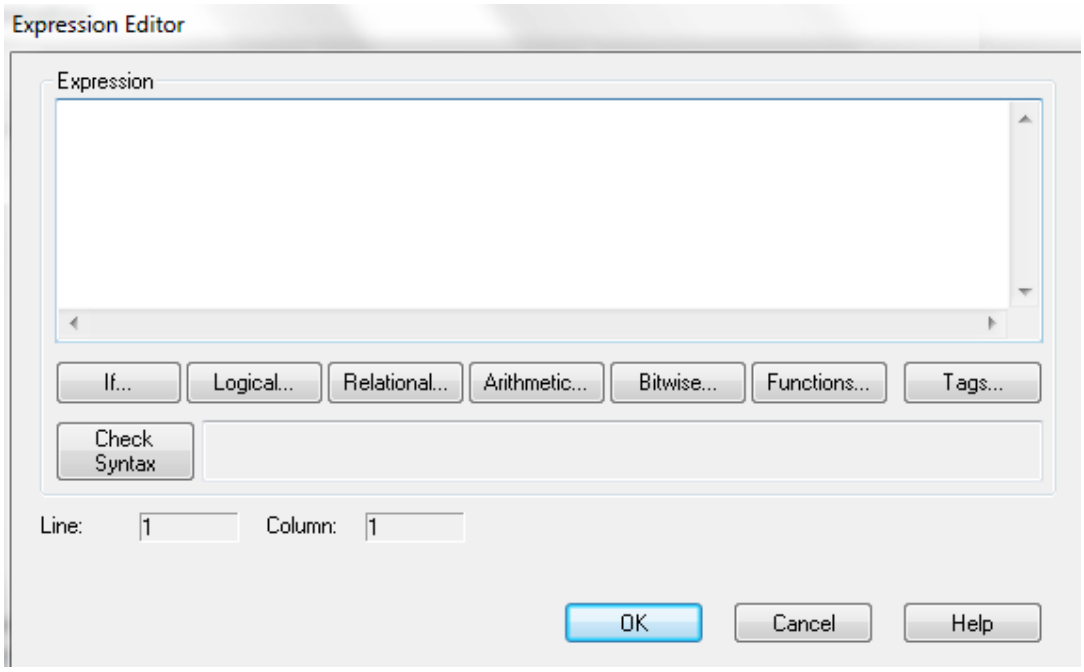


Animation with visibility is available with Allen-Bradley as well. It is shown below with visibility as a property with a tag or an expression with tags available to provide logic for the visibility of a device, in this case, a circle.



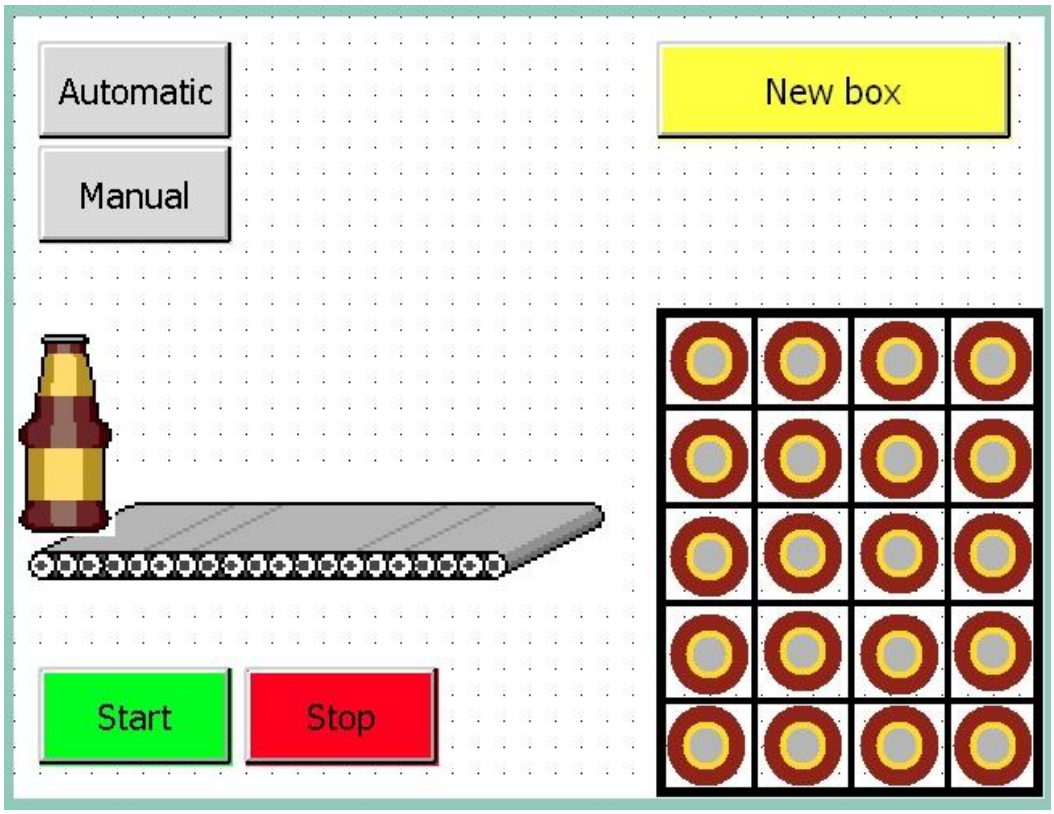


The expression editor gives the ability of the programmer to add logic to select the visibility of an object.

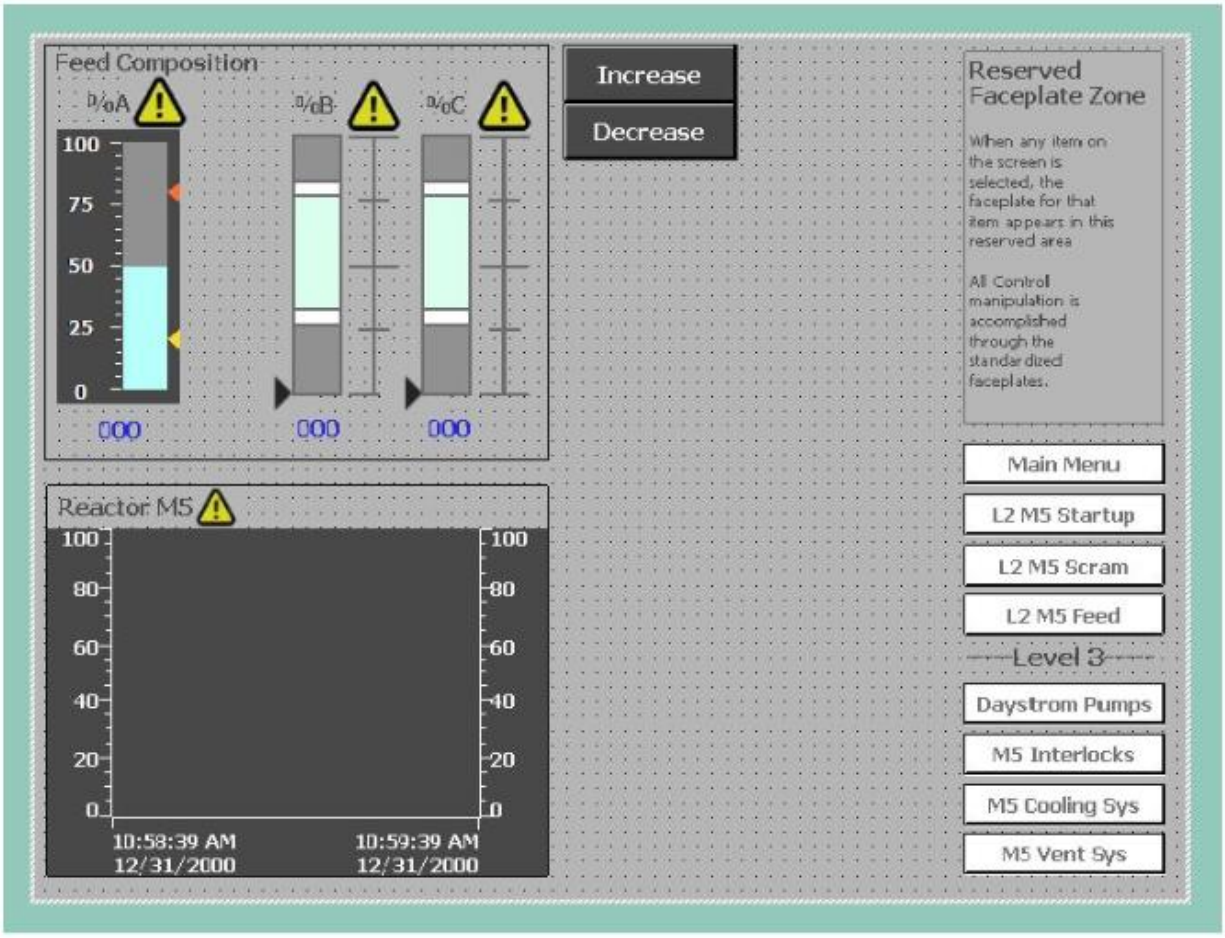


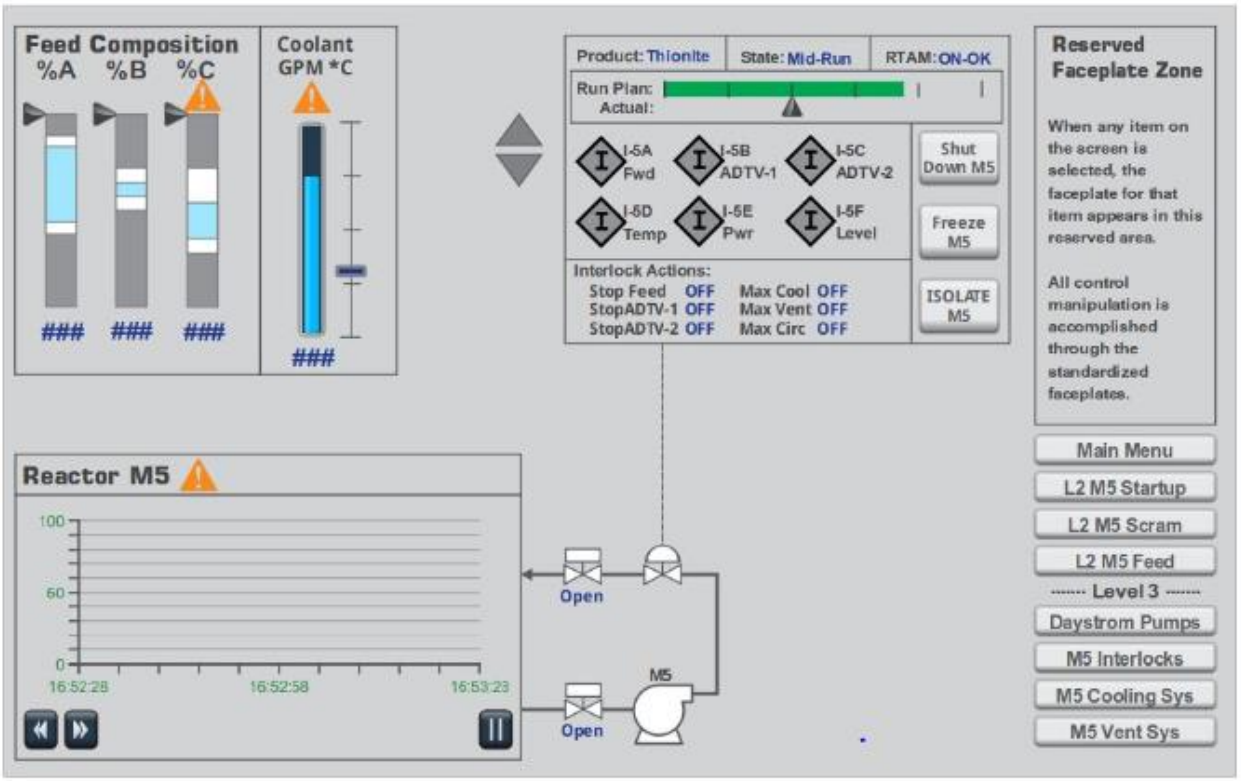
Copy and paste the individual bottles.

At the animation Visibility of both circles decrease the value to by 1.
The last bottle has the value range from 0 to 0.



The following are two graphic examples created by EET student Lee Monday. They give examples of the type of HMI screens discussed in the next section on HPHMI.





How to Improve Plant Operations through Better HMI Graphics

A number of good texts are available for better design of HMI graphics including:

The High Performance HMI Handbook by Bill Hollifield, Dana Oliver, Ian Nimmo and Eddie Habibi

Designing for Situation Awareness (An approach to User-centered Design) by Mica R. Endsley and Debra G. Jones

Human Machine Interface (HMI) Design: The Good, The Bad, and The Ugly (and what makes them so) by Paul Gruhn, P.E.

From Hollifield: "Five areas are primary in the successful design of a good HMI screen system. They are:

- Situation Awareness
- Using Color Effectively
- Interpreting the data
- Depicting Device State
- HMI Display Organization

Situation awareness relates to the goals and objectives of a specific job or function. First of all, designers and engineers form in their heads another mental model of the process than an operator. By understanding how operators select and use goals, designers can better understand how information is perceived. Without understanding the user's goals on Situation Awareness, the information that is presented has no meaning."

Hollifield gives a number of graphic displays of the evolution of the operator interface. The following figure was found until the mid 1980's. These panels were created and manufactured for the plant floor and were extremely difficult to change once deployed.

Fig. 15-17
Example of a Control Wall

More computer driven HMI panels began emerging to replace these panels although the early computer replacements were extremely expensive, especially when compared to today's HMI. Early HMI panels also had the disadvantage of little ability to extract or analyze process data.

From Hollifield: “When the modern systems were introduced, they included the capability to create and display graphics for aiding in the control of the operation. However, there were no guidelines available as to how to create *effective* graphics. Early adopters created graphics that mimicked P&ID or schematic drawings, primarily because they were readily available. The limited color palette was used inconsistently, and screens began to be little more than crowded displays of numbers on a P&ID.”

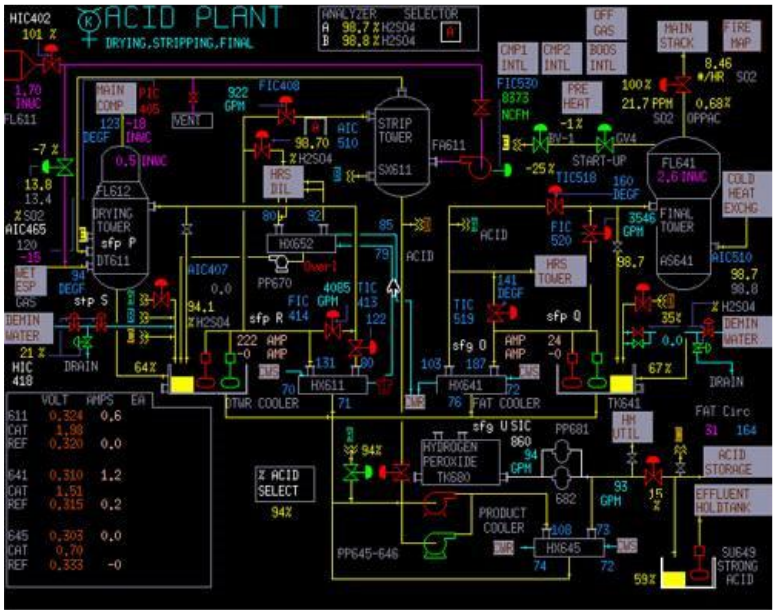


Fig. 15-18
Early Graphic Showing Many Problematic Practices

Early graphics were a stark contrast to the hard-wired panels of earlier times. They were developed primarily from the P&ID drawings of the process but lacked any methods to quickly diagnose problems in the plant. While this style was accepted and promoted, there are severe problems with this design as noted by Hollifield and others. The next few pages outline some of the common mistakes made by continuing to use the design shown above.

With each design, the operator must be considered first. The operator may be a person with little or no process education and may have little or no training. He or she may be on the job only a few days and be required to make quick decisions concerning the operation of the plant. While there are not many examples of plants being destroyed because of operator error, the operation of the plant must be considered from all angles and the HMI panel is the first place an operator or maintenance person should go for help. The following example shows 3-D design but is of little help with actual information.

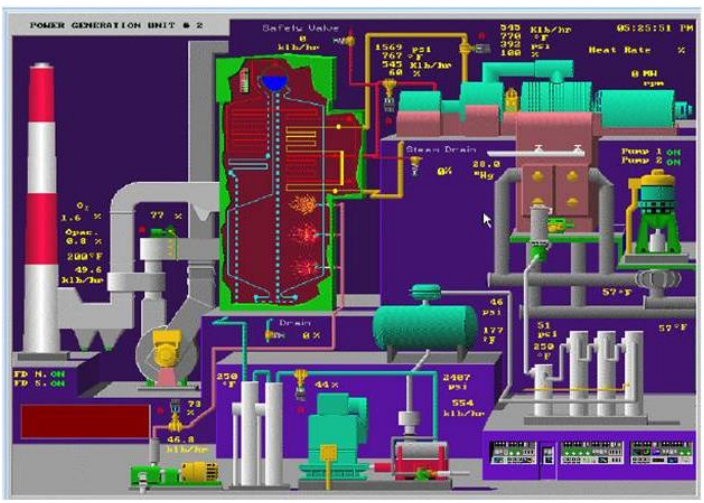


Fig. 15-19
A Flashy Graphic inappropriate for Actual Operational Control

Hollifield points out: “There are no trends, condition indicators, or key performance elements. You cannot easily tell from this graphic whether the operation is running well or poorly. That situation is true for more than 90 percent of the graphics used throughout industry today because they were not designed to incorporate such information. Instead, they simply display dozens to hundreds of raw numbers lacking any informative context.”

Justification for HMI Improvement

An industry with a great interest in excellent HMI displays is the avionics industry. The following is a flight simulator for a small plane. This training is of vital importance prior to the pilot student getting into a plane.

Fig. 15-20
Garmin G2000
Avionics Package
in a Small Plane

The following shows an example of a great deal of information yet no real information about the process. This is not what to do!

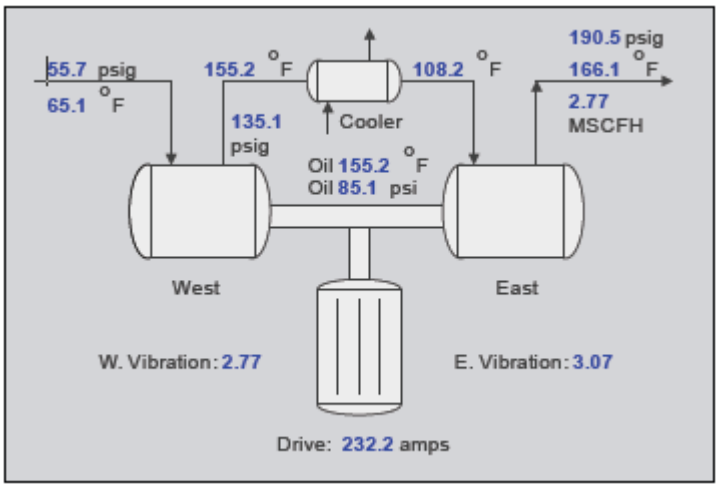


Fig. 15-21
All Data, No Information

The same information as depicted in Fig. 15-21 above is displayed in the graphic below in Fug, 15-22. It shows data pertinent to the process with enough information about those values to make logical judgments about the process. The data is mostly analog.

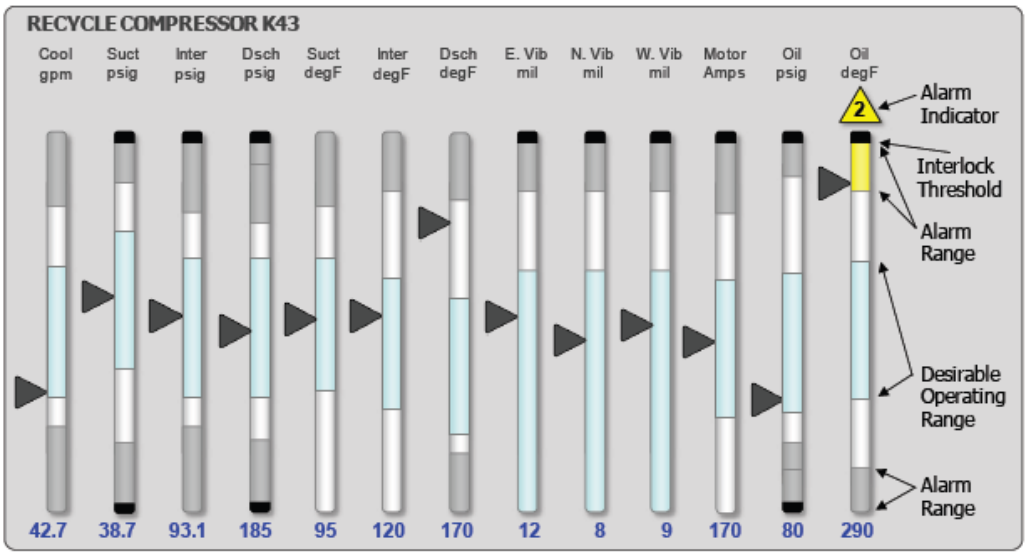
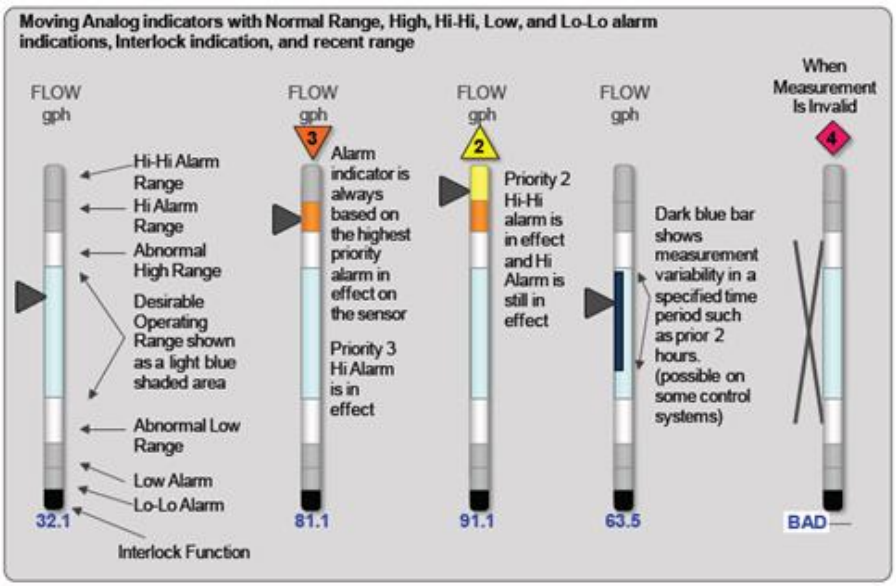


Fig. 15-22 Analog Depiction of Information



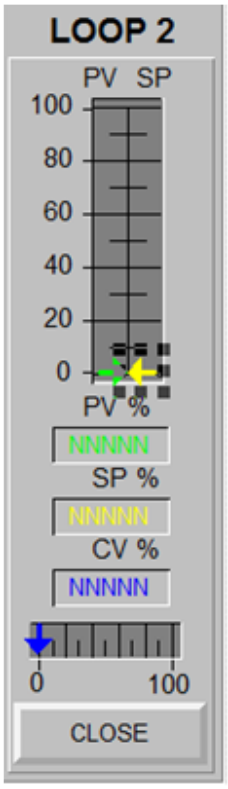


Fig. 15-23 Further Explanation of Moving Analog Indicators

In the figure above, PID controllers are shown with active values displayed. Pointers show the actual value on a sliding scale. Alarms are shown with color and verbiage. The PID faceplate at right is the more classical PID graphic. Either type of display is acceptable but fundamental values are necessary in either. The written warnings at left are desired with either display.

From Hollifield again, "*Color, by itself, is never used as the sole differentiator of an important condition or status*

Most graphics throughout the world violate this principle. A color palette must have a limited number of distinguishable colors used consistently. Bright colors are primarily used to bring or draw attention to abnormal situations, not normal ones. Screens depicting the operation running

normally should not show brightly saturated colors, such as bright red or green pumps, equipment, valves, and similar items."

Tables can be useful for display of large amounts of data. The HMI display should be used to instruct or teach the operator as changes in the process necessitate action:

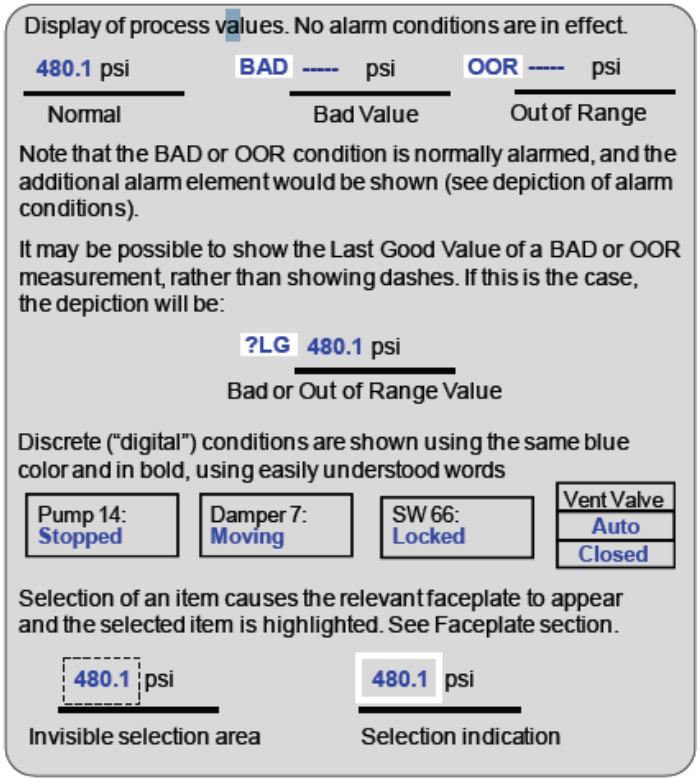


Fig. 15-24
Depicting Process Values

A good example of depiction of alarms is the following:

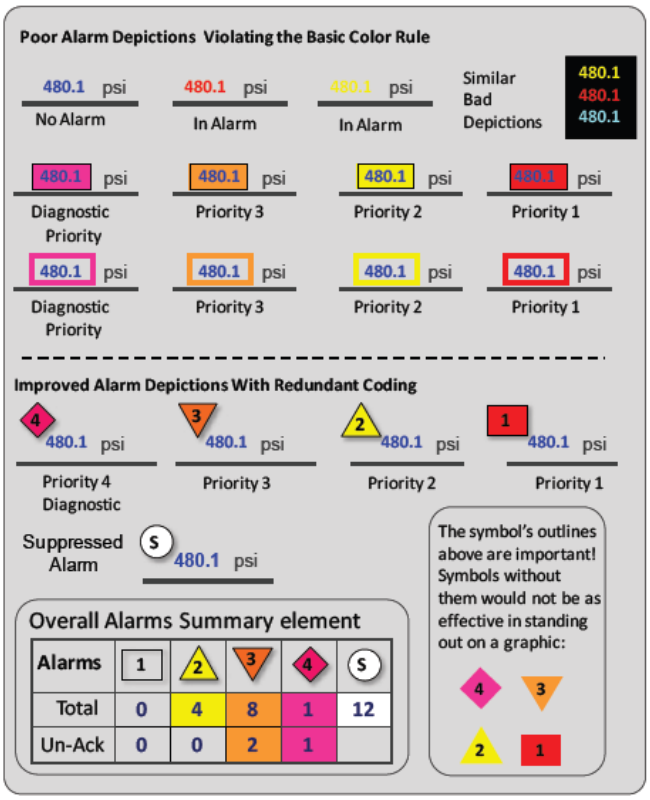


Fig. 15-25
Depiction of Alarms

For linked alarms, the following is a good example:

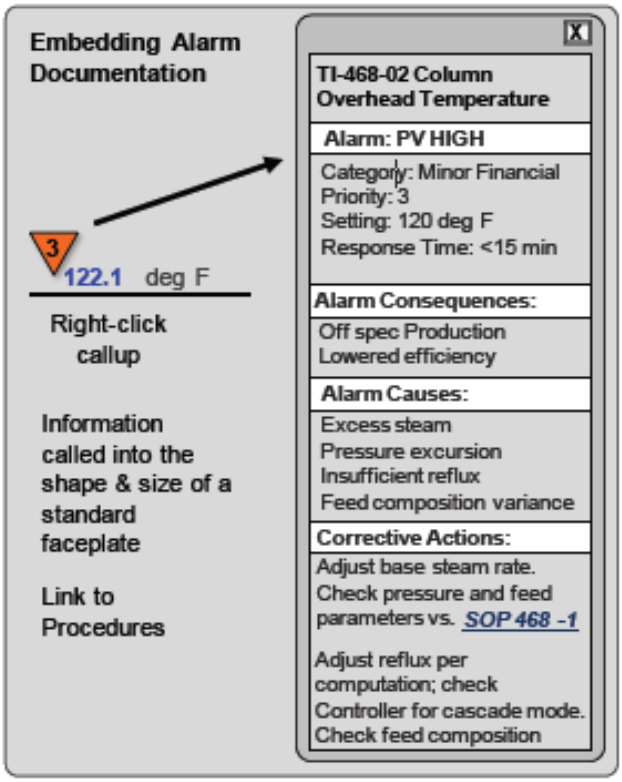
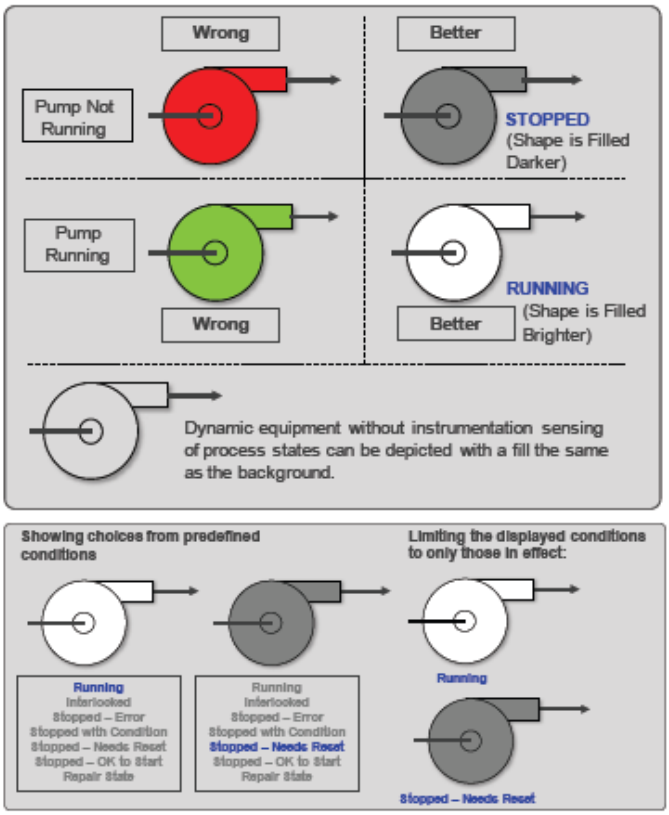


Fig. 15-26
Linked Alarm Information

Depicting Dynamic Equipment

Use of color for running conditions of motors is not good. The following shows the better use of color brightness for running vs stopped motors.



The relative brightness of the object shows its "ON-oFF" status, as does the use of a process value *word* next to it. Equipment items brighter than the background are "ON" (think of a light bulb inside them). Items darker than the background are "OFF." If equipment has no status that is sensed by the control system, but is desired on the graphic anyway, it is shown as transparent to the background color. The status word can indicate several conditions, as shown. Remember, if any of those are also alarm conditions, the separate alarm indicator will appear next to the equipment when it is in an alarmed state.

Fig. 15-27 - Depicting Status with Redundant Coding and Proper Color Usage

Bars vs. Pointers

Pointers are better for showing analog values rather than bar graphs. The following point out this advantage.

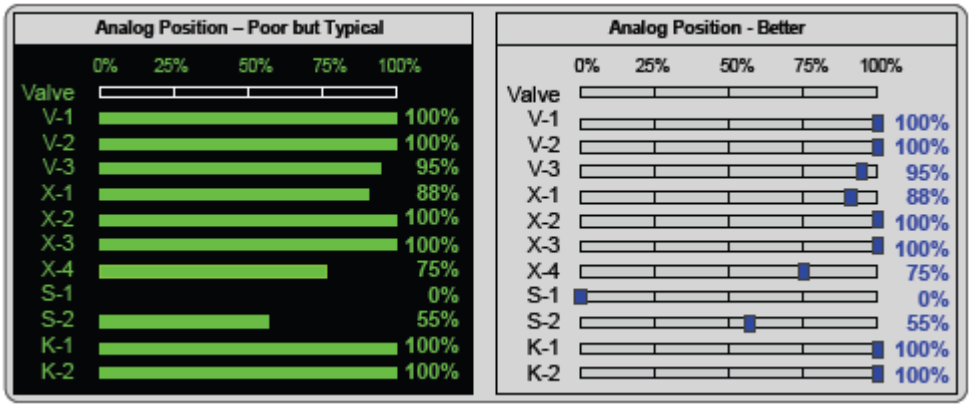


Fig. 15-28
Bars vs Pointers

Depicting Level Indication

The following show a progression from very poor to very good depiction of level control in a tank.

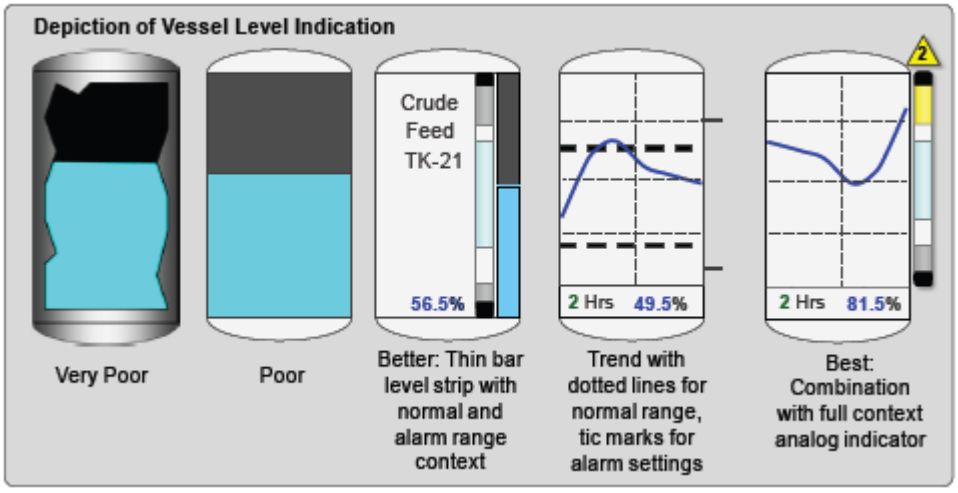


Fig. 15-29
Vessel Levels

Depicting Control Valves and Shutoff Valves

The following gives examples of control valves. Closed position is shown in black and open in white

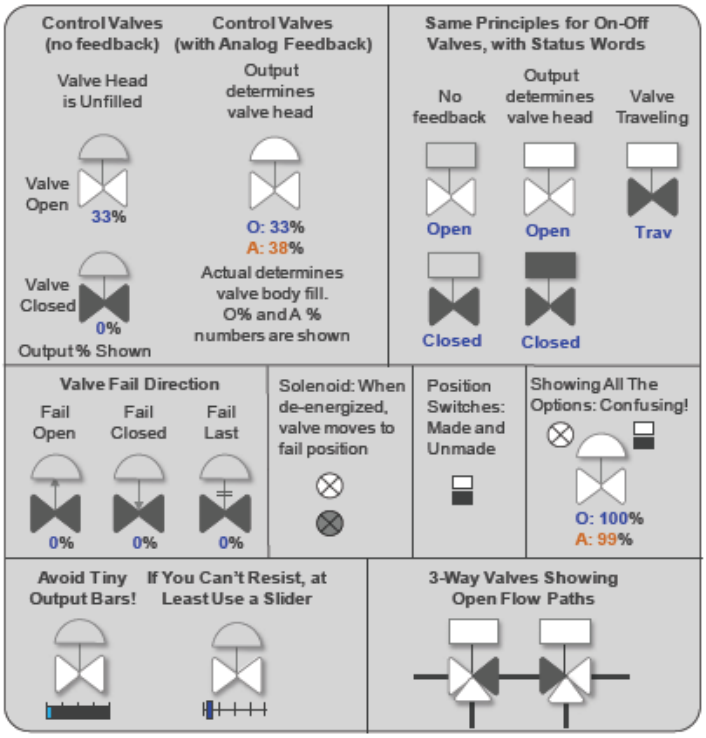


Fig. 15-30
Control and Automated Block Valves

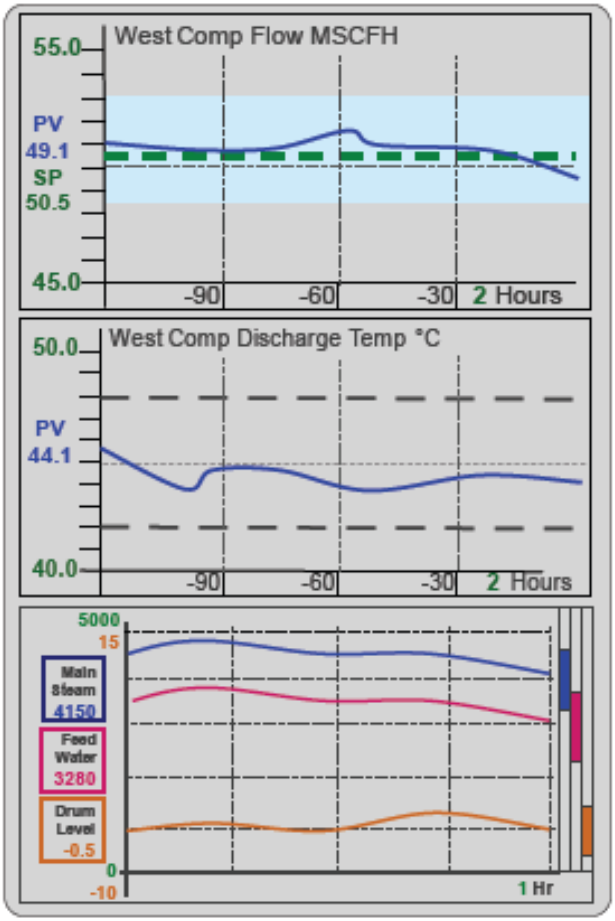


Fig. 15-31
Trend Depiction of Desirable Ranges

Trends should be **embedded** in the graphics and appear, showing proper history, whenever the graphic is called up. This is generally possible but is a capability often not utilized. Trends should incorporate elements that depict both the normal and abnormal ranges for the trended value

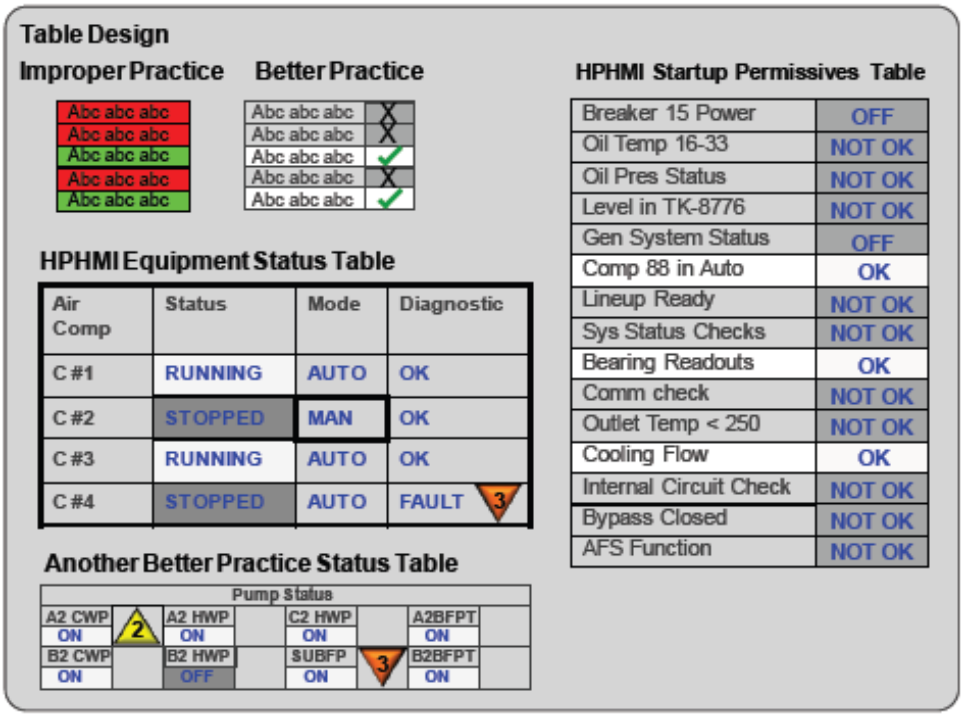


Fig. 15-32
Tables and Checklists

Depicting Shutdown Activation

"Operators must have the ability to shut down operating equipment manually and quickly. However, when an important action with significant consequences is based upon operator input, the input should have a confirmation mechanism that avoids inadvertent activation. The "cancellation" option should be consistently implemented.

It should never be possible to make a single selection on a screen that results in an inadvertent shutdown. A "Shutdown button" should call up at least one, and perhaps two, layers of confirmation before it is possible to actually cause such a significant event."

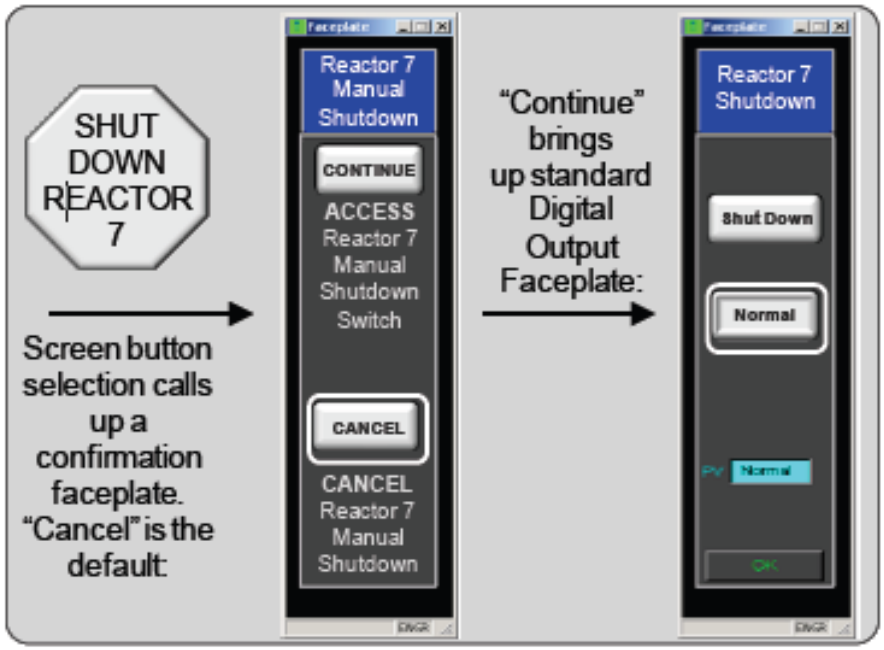


Fig. 15-33
Layers of Confirmation

Depicting Interlock Functionality

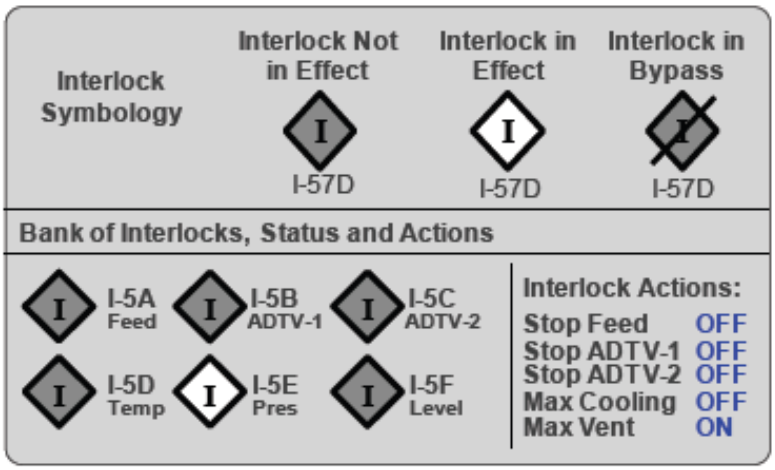


Fig. 15-34
Interlock Symbology Example

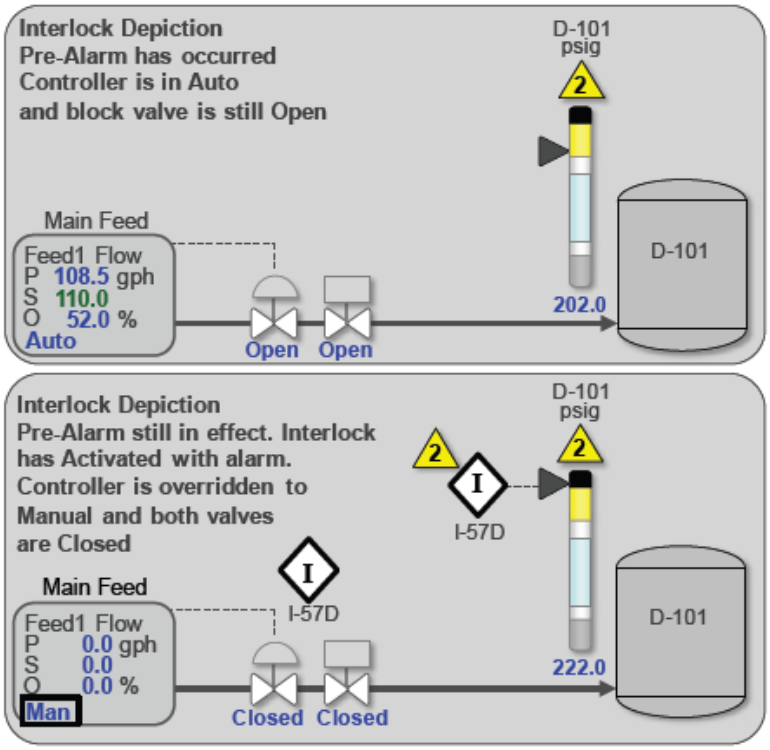


Fig. 15-35
Interlock Before and After Activation

West Compressor Interlock Initiators	W-1A Mech	W-1B Flow	W-1C Pres
Comp in Overspeed	NO	1 Stg Pres is High	YES
Winding Temp is High	NO	2 Stg Pres is High	YES
Vibration is High	NO	Suction Pres is Low	NO
Oil Pres is Low	NO	Total Flow is Low	NO

West Compressor Interlock Actions			
W. Comp Shutdown	NO	Flow Cascade to Manual	YES
Inlet Block Valve	OPEN	Maximum Flow Bypass	OPEN
Outlet Block Valve	OPEN	Maximum Cooling	NORM
E. Comp Override to 100%	NO	Winding Purge	OFF

Fig. 15-36
Interlock Diagnostic Table

Compressor 43 Shutdown Initiators		
Overspeed		OK
West VIB >25	X	1.5 First Out
West Bearing Temp >220		160.3
East VIB >25		1.3
East Bearing Temp >220		155.3
Suction Pressure <20	X	12.3
Discharge Temp >240		148.7
Oil Pressure <10	X	3.3

Fig. 15-37
Shutdown Initiator with First-Out

Startup Map

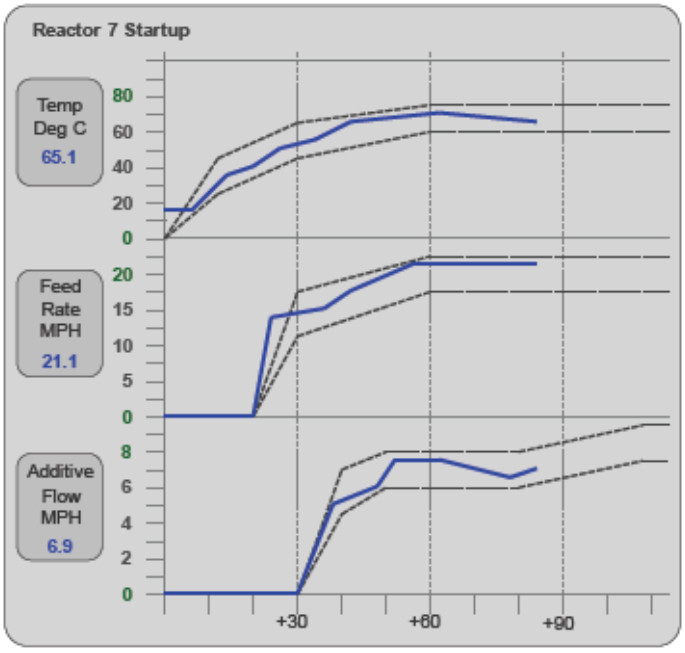


Fig. 15-38
High Performance Element
Designed for Startup Use

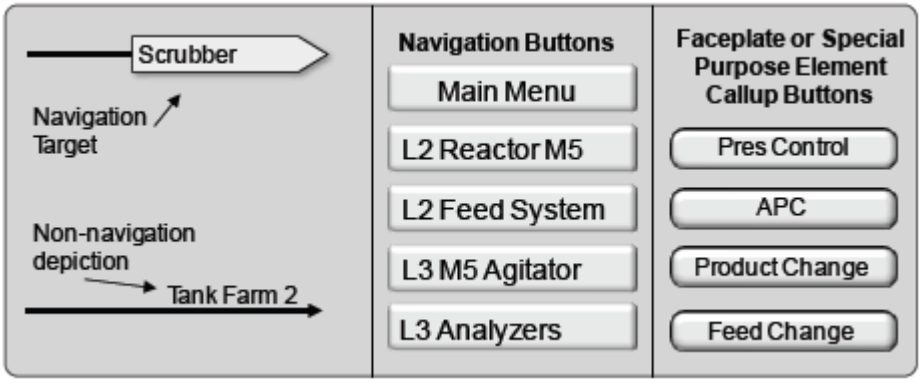


Fig. 15-39
Navigation Buttons
and Targets

Display Hierarchy

Displays should be designed in a hierarchy that provides progressive exposure of detail.

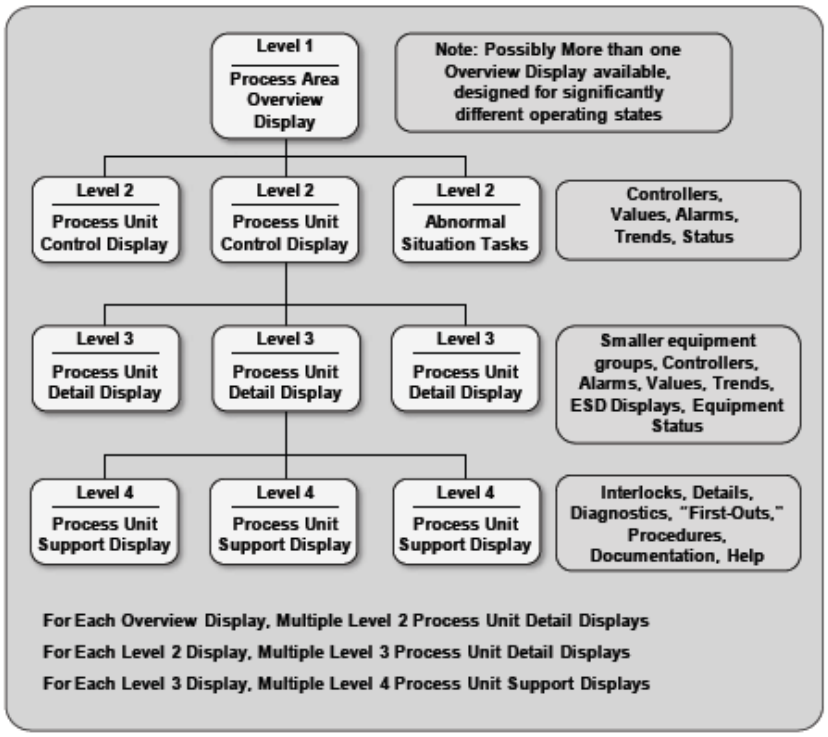
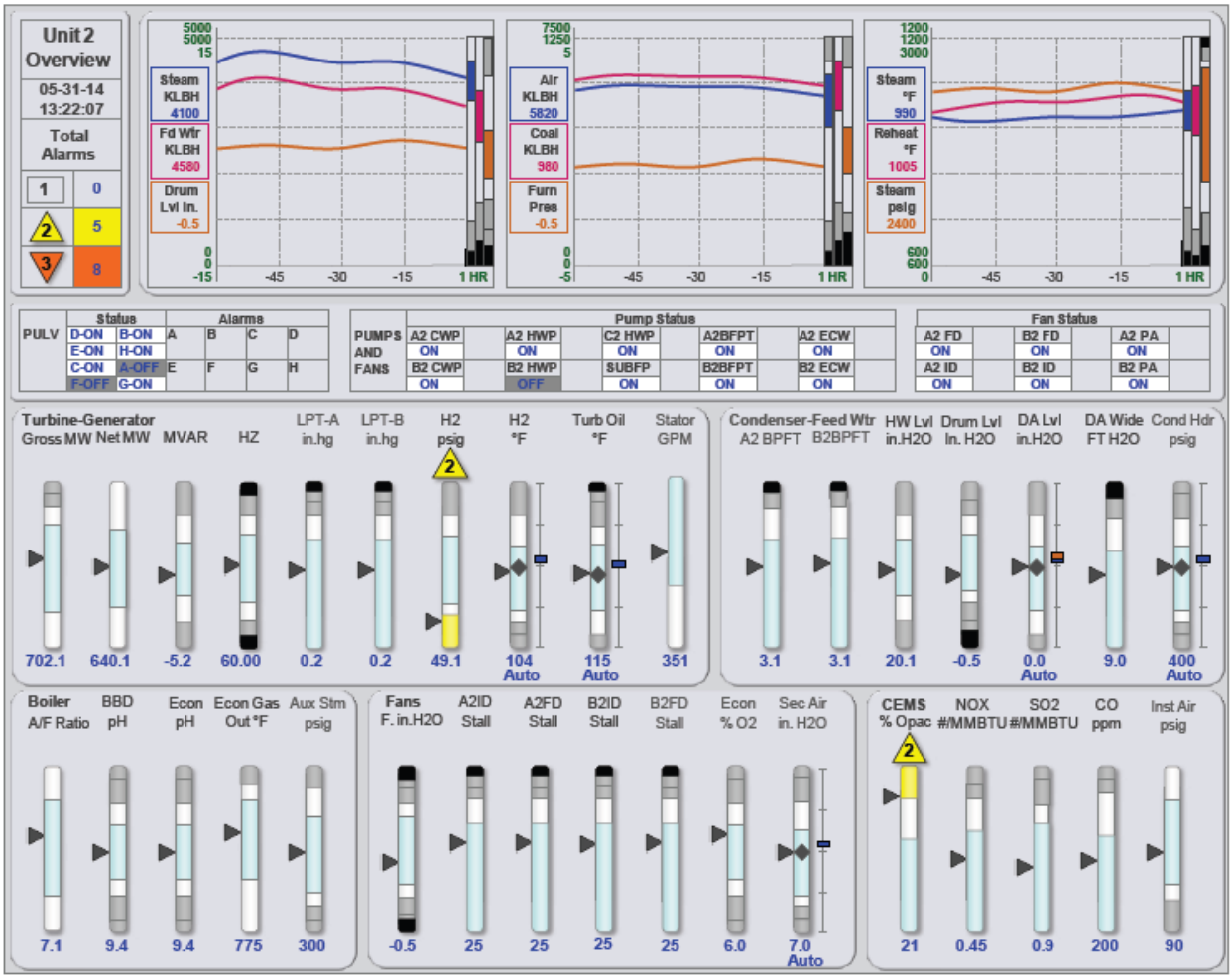
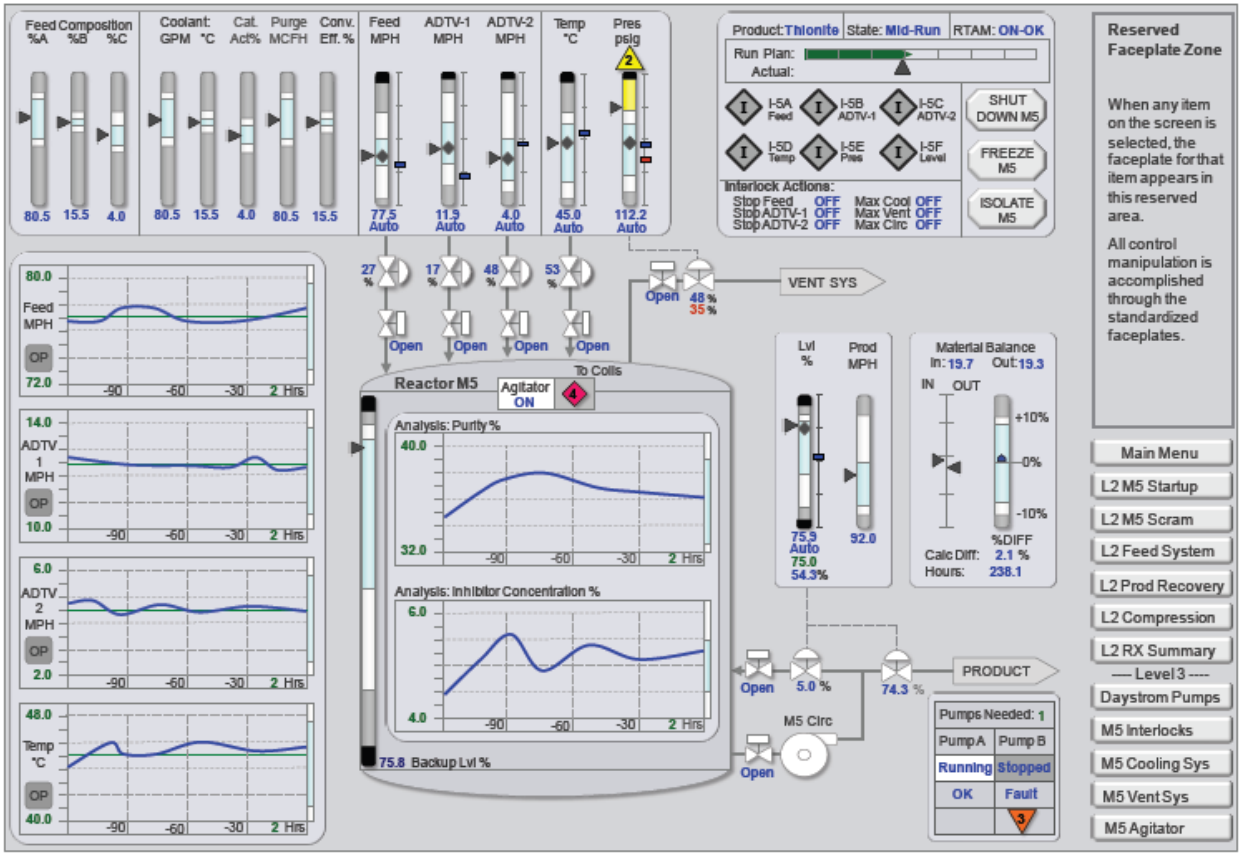


Fig. 15-40
High Performance HMI Display Hierarchy

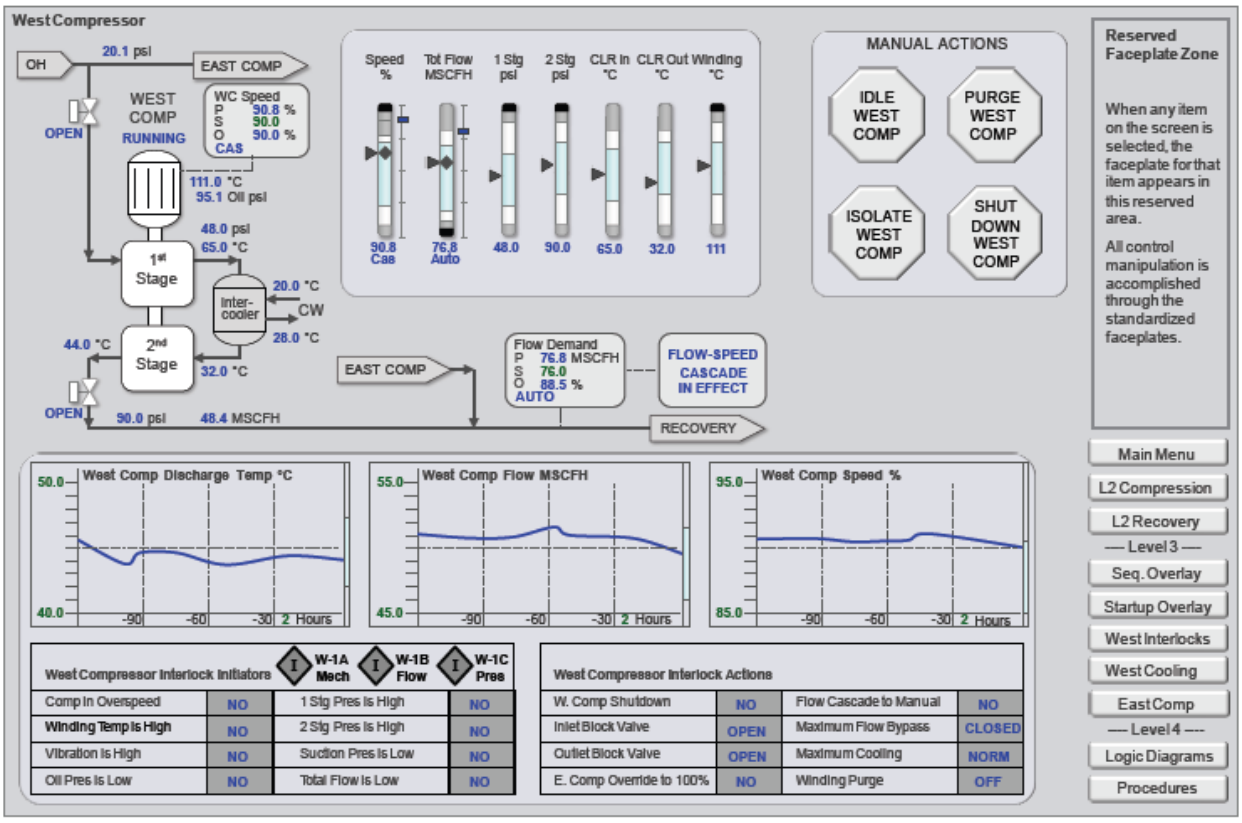
Level 1 – Operation Overview



Level 2 – Unit Control



Level 3 – Unit Detail



(Level 4 is used for support and diagnostic displays.)

OPC and Visual Basic

OPC is short for OLE for Process Control. OLE is short for Object Linking and Embedding. OPC strives to connect industrial automation with software programs (sometimes referred to as enterprise systems) to share data. OPC is an open system with shared standard approaches. Currently seven standards comprise the OPC system. OPC Foundation is the organization to oversee the adoption and creation of these standards.

OPC is a program that works with OLE (Object Linking and Embedding) a technology developed by Microsoft for the purpose of embedding and linking to documents. OLE stands for OLE for Process Control. Included in OPC are devices that provide aid to the programmer in areas such as:

- **Alarms and Event**

- **Redundancy:** Industrial applications frequently require high availability and reliability that can be easily achieved by implementing communication redundancy

- **Client Server Architecture:** The client/server nature of OPC enables users to architect connectivity solutions that would previously be prohibitively expensive.

- **Historical Data Access:** OPC Historical Data Access (OPC HDA) specification is used to archive and retrieve process data. Also included are trends reports using the OPC HDA client applications.

DDE and OPC are integrated into the Allen-Bradley product through RSLinx. See the opening tabs for these applications in RSLinx below:

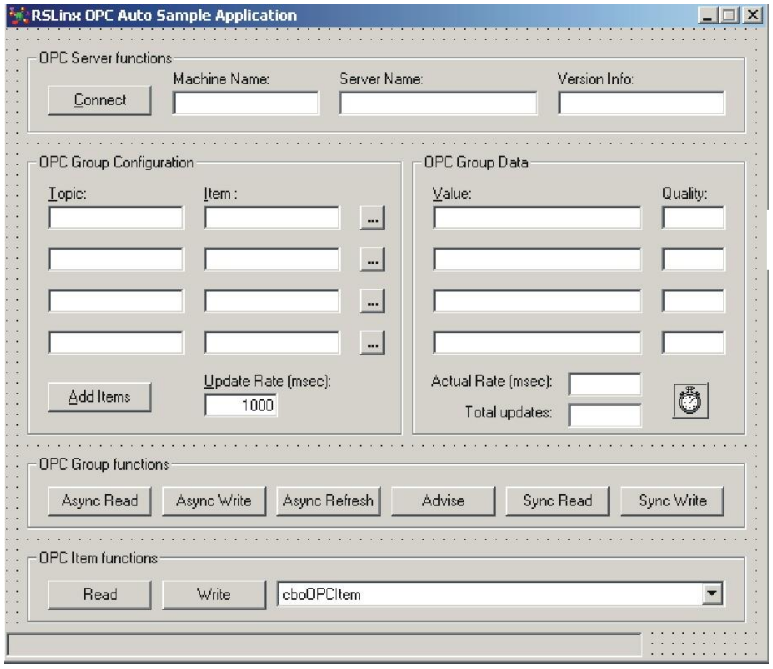


Fig. 15-41

Microsoft's Access and Crystal Reports are examples of the power of using OPC with Visual Basic.

Graphic Standards From Windows Standards

Graphics come in many flavors but not all file formats are suitable for all purposes. How do you know which is best? Some standards exist for a specific company. Other standards for graphics are general and used by most. Some of these are:

Practical Design of Logic with HMI

From early in the chapter, A-B's design allows for a Push Buttons with the selection Multistate. The memory circuit can be turned on or off from the multistate button. The better memory circuit can also be turned on or off from the program. The memory circuit then must be able to report to the HMI the present state. The program for this function is left as an exercise.

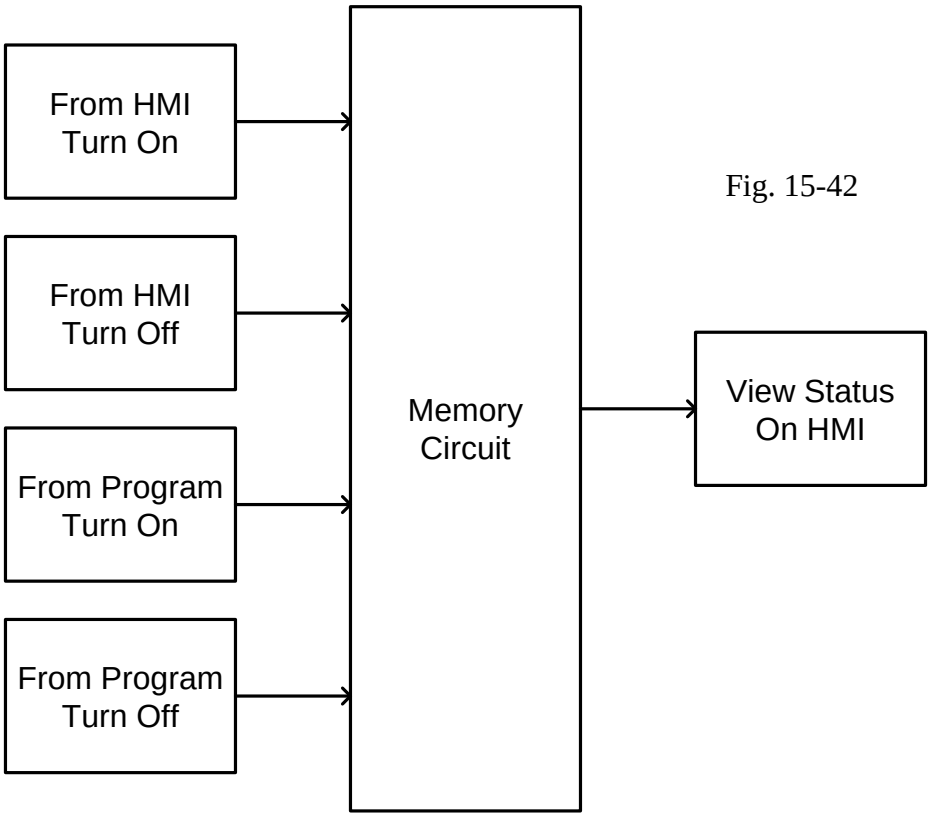


Fig. 15-42

The use of multistate buttons to provide this logic is useful but not necessary. The use of a single button on the screen is an advantage in that one button can be used to turn on the memory, turn off the memory and display the present state of the memory. All three functions can be accomplished with a single button. All the time in advertising, we hear the ad for buy one, get one free or the 'two-fer' ad. This is a real 'three-fer'. Buy one button and get the 'start' button, the 'stop' button and the 'indicator' light in the same button.

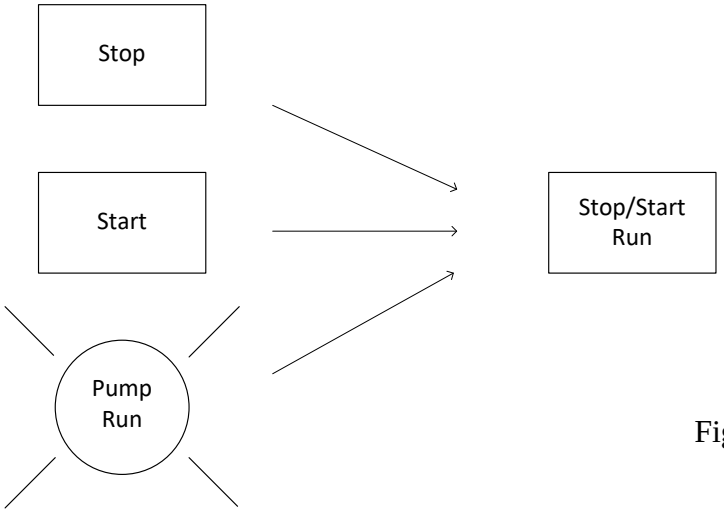


Fig. 15-43

Where to Put the Logic

The following conveyor system has five outputs, lights for percent complete of packages going down conveyor 1 to conveyor 2. Write a program to turn on these lights based on the fact that packages must pass photo-eye 1 to enter the storage area and pass photo-eye 2 to exit. This program was solved in Ch. 8 using greater/less-than statements and discrete outputs. It is possible now to turn on these outputs in the HMI program with no statements in the Ladder other than the Up-Down counter.

You may find it easier or better to provide the logic in the HMI or Ladder. There is usually a preference in most companies for one or the other or you may decide for them.

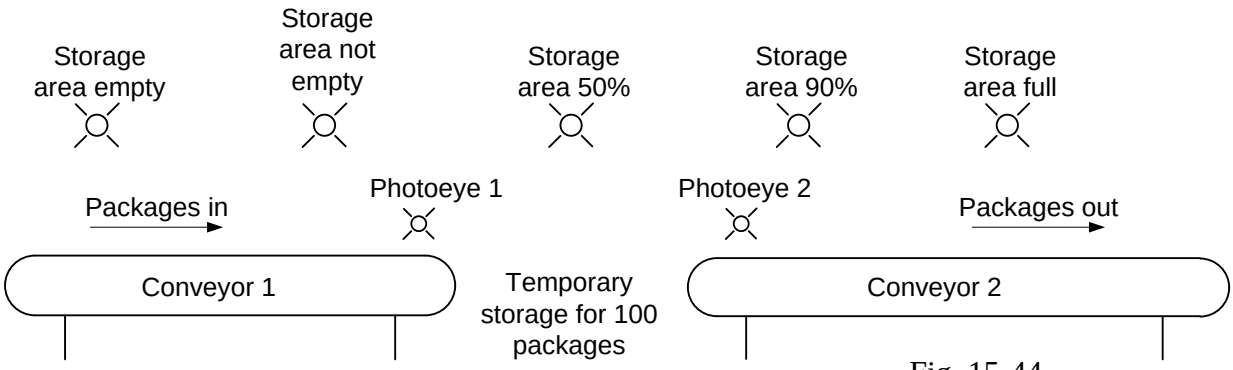


Fig. 15-44

Consider Multi-State Indicators for the application above using RSView Studio as an example:

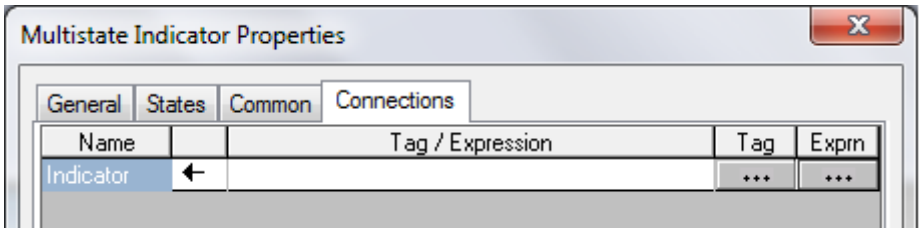


Fig. 15-45

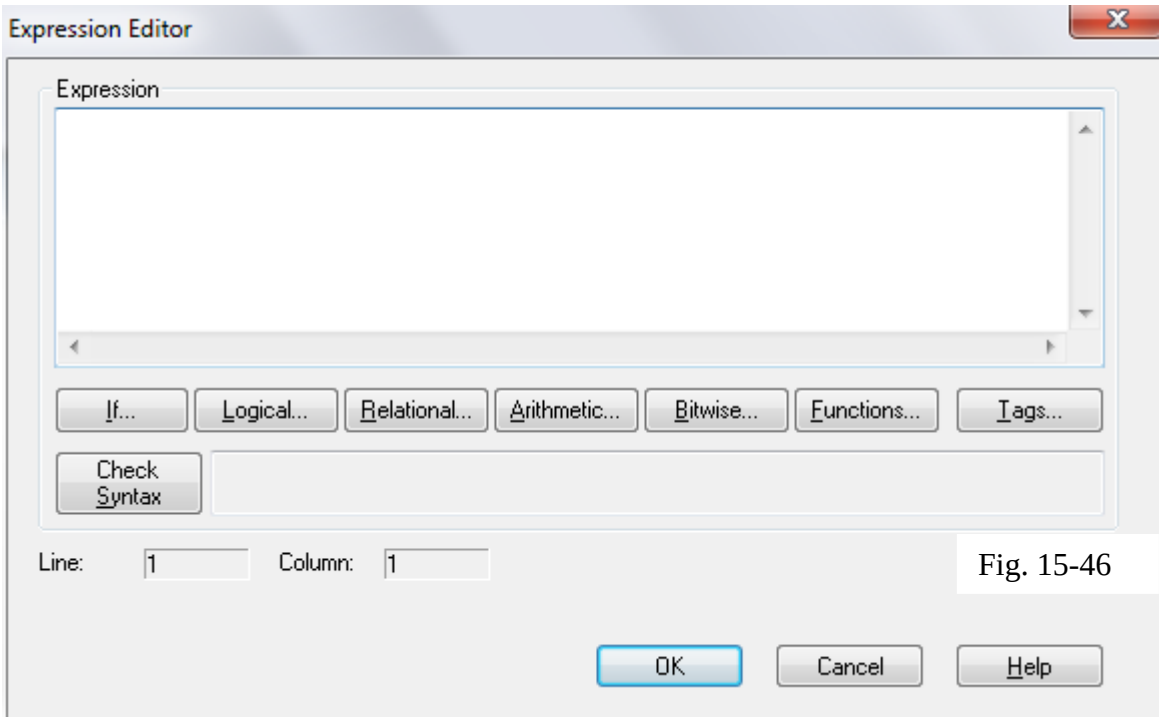


Fig. 15-46

For these examples, assume tag1 = 5 and tag2 = 25.

Expression	Returned Value
tag1	**5**
tag1 + tag2 (arithmetic operator)	**30**
~tag1 (bitwise operator)	**-6**
SQRT(tag2) (mathematical function)	

Example: Expressions that return true/false values

Expression	Returned value
tag1 > 20 (relational operator)	1 (true) if tag1 is greater than 20 0 (false) if tag1 is less than or equal to 20
Industry\Valve AND Municipal\Valve (logical operator)	1 (true) if both valves are open 0 (false) if one or both valves are closed

Example: Controlling visibility with If-Then-Else logic

To create a graphic object that is to be visible only when *tag1* exceeds a specified value

1. Draw the object.
2. In the Visibility animation dialog type the expression:

If (tag1 > 55) Then 1 Else 0

3. Specify that the object is to be visible when the expression is true.

From 1973, an HMI That Worked After Months of Development

In 1973, I was intensely involved in a project involving an HMI for three stations for a glass manufacturing line. The program was written entirely in assembler on a 16 k machine (IBM System 7). The HMI portion was to be six screens for the entire line including the three stations. What happened to that project was the complete explosion of need for a manual back-up that could over-ride the automatic system originally envisioned. The only response was to create a manual back-up system that expanded the original six screens to 66 screens. Along the way, a complete diagnostic system was also developed since electricians and operators could not be expected to keep the line operational without good data concerning the sensors on the machine and which if any were not operating correctly. The eventual HMI program consisted of about 23k assembly statements found in 66 individual over-lay programs running in a 384 word over-lay area of the computer.

The program was a tree-designed system with operators being asked a series of choices from which an operator successively moved further out a branch of the tree. In the end it worked and was accepted by the operators and plant engineers. It took way too long and enveloped my time for more than a year. Hopefully the lessons learned from experiences such as these would be helpful for future engineering efforts.

One of the two learned was the conflict between production and the hypothetical original design that insisted on complete automatic control. The manufacturing environment was not ready for the complete automation of the theorist. If a production supervisor found it more advantageous to have the destination of glass be on a take-off point on the right side instead of the left, he wanted that flexibility. The automatic system did not give that kind of flexibility and he refused to use the system until developed to his needs. The other lesson was that of complete diagnostic reporting. The operation was not going to succeed until every switch was operational (for glass tracking) and these switches were constantly falling into disrepair due to the moving glass. It was only after a dedicated line on each display showed the active status of any broken switch and its exact location was the tracking system allowed to function.

From 1998, an HMI System Divided

In 1998, I worked on a system for a steel company that saw the dividing of the project into multiple assignments with four engineers working simultaneously on two PLC/HMI programs. The PLCs were Modicon and the HMI was Wonderware. In this project I was responsible for the PLC program for process control. My counterpart worked on the HMI portion from India. Two individuals programmed the steel movement portion of the project from their office in Toronto, Canada. The most difficult portion was the HMI portion for the steel movement. It involved a number of rules for the handling of the steel. These rules were total written from the Wonderware software platform.

The rules for the handling of the steel were similar to those of a baker with an oven big enough to handle multiple cakes with various baking times and temperatures. If a cake was in the back

of the oven and was ready to come out, could a new cake be inserted in its place given the time over-lap of a second or third cake presently in the oven or ready also to be placed in the oven? The rules allowed for proper bake time and temperature for each individual cake while controlling the temperature of the overall oven.

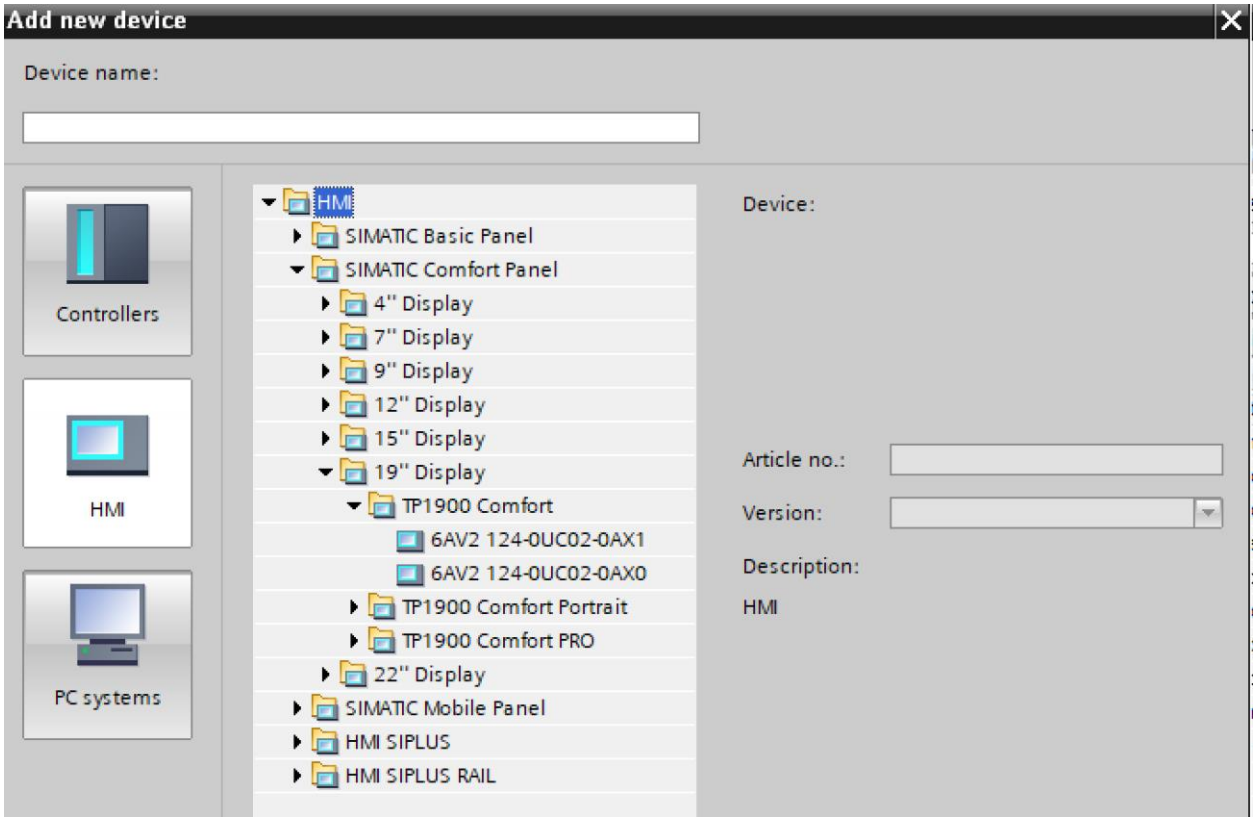
From an article in the dmcinfo website, we see an intriguing topic first introduced last chapter:
“

Multiplexing Arrays of UDTs in TIA Portal V14
By Jason Mayes
Dated 12/06/2016”

Also, we are introduced in the above article to an open library for Siemens. It is described in the following:

“The Siemens Open Library was developed by DMC, Inc. over several years. Through a joint collaboration with Siemens Industry, DMC documented the library and released it as an Open Source Library open for anyone to use and distribute. The Library is Open Source and will allow for users to contribute content to help enhance the overall availability of functionality to the greater Siemens User Community.”

With the introduction of Siemens’ version 17, we now have access to the more sophisticated Comfort Panel software. With this comes the ability to write code behind a button (or anywhere an event occurs). The following example shows this capability.



Inclusion of a button starts the process: Fig. 15-47

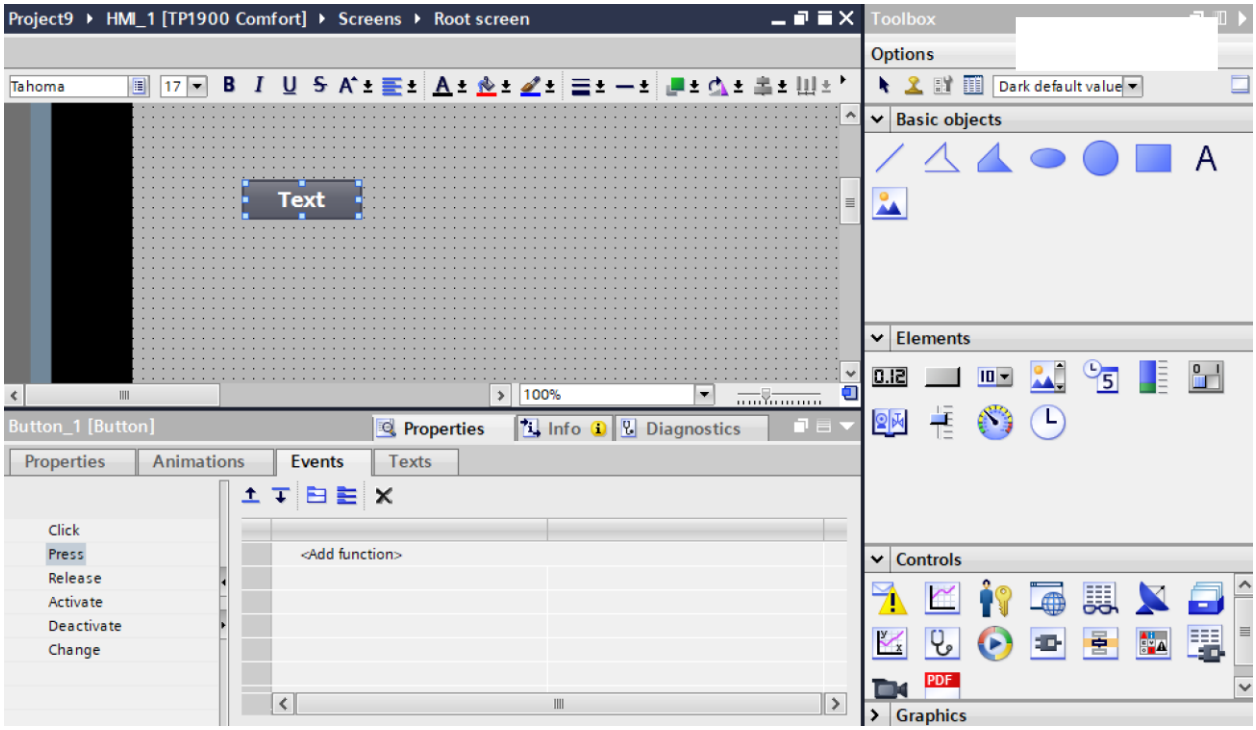


Fig. 15-48

Under ‘event’ and ‘other function’ we find ‘StartProgram’. This is the function we want to use to allow large amounts of programming in the HMI.

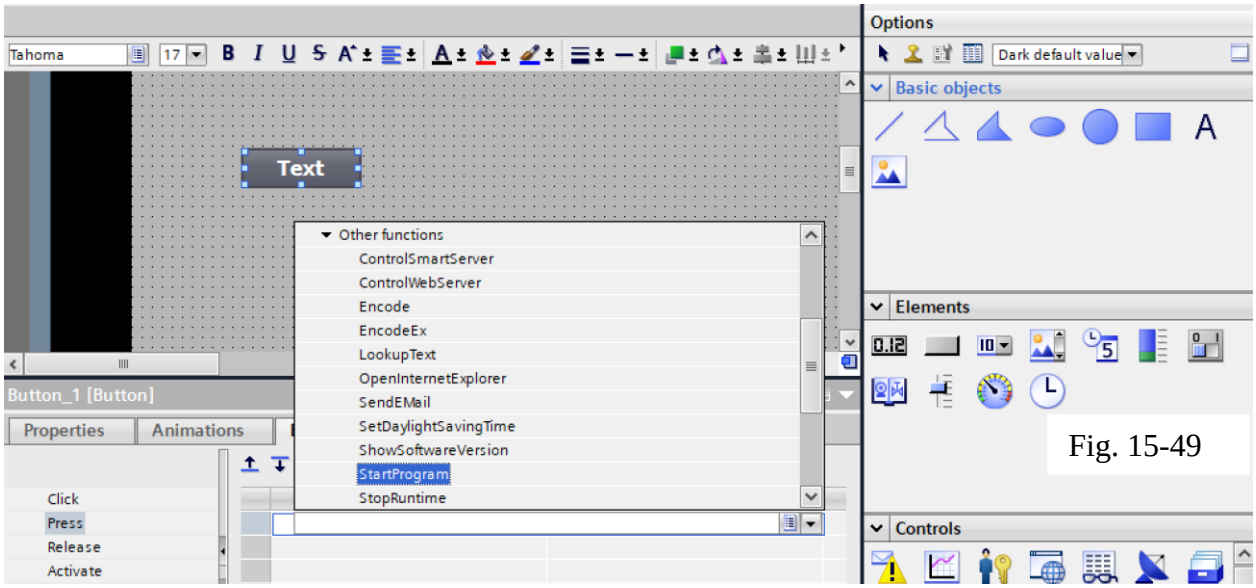


Fig. 15-49

Here we are being asked for the program name, the final step in linking the button with a program. This is a powerful enhancement to the HMI programming function.

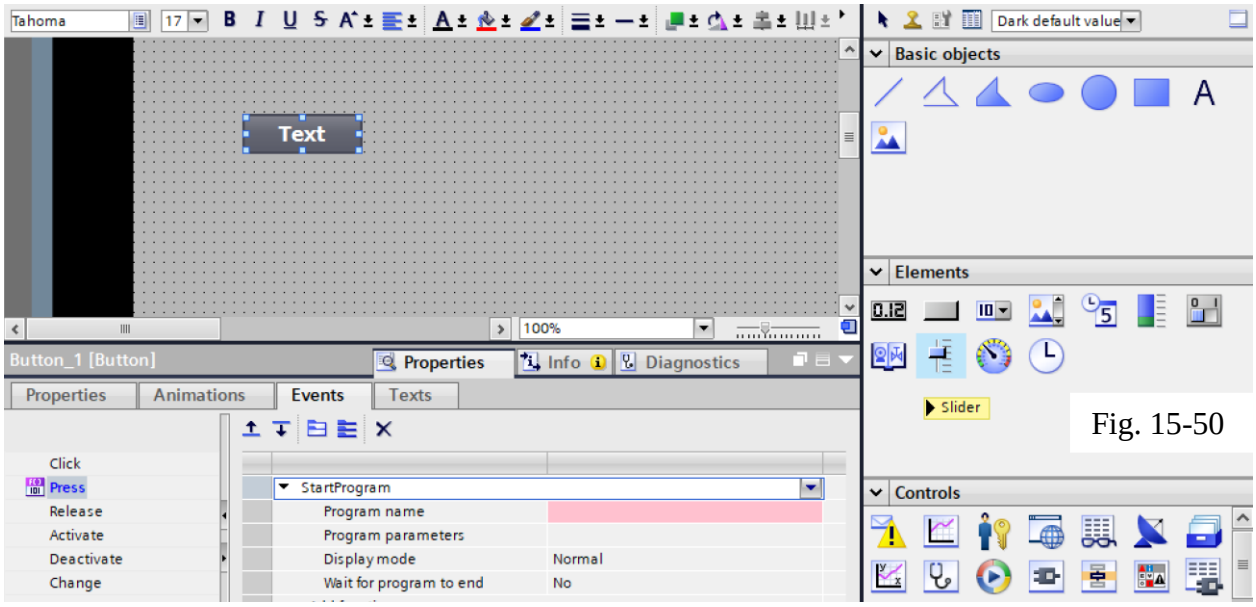


Fig. 15-50

What is next is up to you, the student and user. The answer is – anything – whatever you would like to do. You now have the ability to write extensive programs in the HMI programming portion of the TIA programming environment. This is encouraged – and a lab is given for you to begin.

A great place for ideas on Advanced HMI Programming is the following:

https://www.dmcinfo.com/latest-thinking/blog/articletype/categoryview/categoryid/66/hmi-and-scada

The following pages are from the Siemens Text:
Programming Guideline for S7-1200/1500 Entry ID: 81318674, V1.6, 12/2018
They summarize changes and upgrades to the Siemens Portal Language from Version 14 – TIA and later:

“

2.13 Internal reference ID for controller and HMI tags

STEP 7, WinCC, Startdrive, Safety and others integrate into the joint data base of the TIA Portal engineering framework. Changes of data are automatically accepted in all the locations in the user program, independent from whether this happens in a controller, a panel or a drive. Therefore no data inconsistencies can occur.

If you create a tag, the TIA Portal automatically creates a unique reference ID. The reference ID cannot be viewed or programmed by you. This procedure is internal referencing. When changing tags (address), the reference ID remains unchanged.

2.13 Internal reference ID for controller and HMI tags

In the figure below the internal reference to the data is displayed schematically.

Figure 2-23: Internal reference ID for PLC and HMI

PLC1 **HMI1**

PLC Symbol name	Absolute address	Internal PLC reference ID	Internal HMI Reference ID	HMI Symbol name	Access mode	Connection with PLC
motor1	I0.0	000123	009876	motor1	<symbolic access>	PLC1_HMI1
valve2	Q0.3	000138	000578	valve2	<symbolic access>	PLC1_HMI1

Note

The ID is changed by ...

- renaming tag.
- changing type.
- deleting the tag.

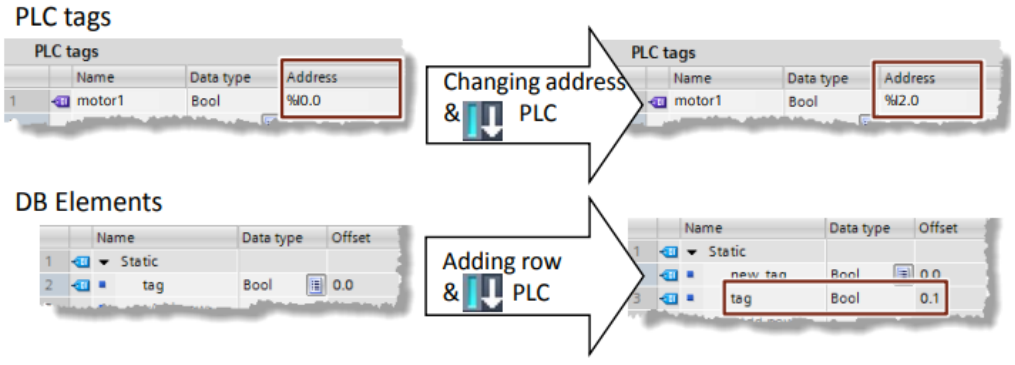
Advantages

- You can rewire tags without changing internal relations. The communication between controller, HMI and drive also remains unchanged.
- The length of the symbolic name does not have an influence on the communication load between controller and HMI.

Properties

If you change the addresses of PLC tags, you only have to reload the controller since the system also addresses the system internally with the reference IDs. It is not necessary to reload the HMI devices (see Figure 2-24: Changing address or adding row).

Figure 2-24: Changing address or adding row



6 Automatically generate visualization using the user program

6.1 Introduction

As of TIA Portal V14, you can use the SiVArc (SIMATIC Visualization Architect) option package to automatically generate the visualization of your plant from a visualization library and the user program in the controller.

In this chapter you optimize your user program for use with SiVArc.

Advantages of generation with SiVArc

- Automatic generation of the visualization with process connection
- Standardization of user interfaces
- Simple and consistent adaptations to operating screens

Requirements

The basic requirement for the use of SiVArc is a high **degree of standardization** of your plant. Modularizing the system into individual function groups has the advantage that SiVArc can use these function groups to generate and interconnect the operating screen from existing library screens. Standardization of the interfaces contributes to efficient work and automatic generation of the visualization.

Compliance with the general recommendations of this programming guide will help you.

Note

SiVArc is an interface topic between HMI and controller. This section looks at SiVArc from the control side.

The following links will give you an insight into the functionality of SiVArc:

Application example "SiVArc Getting Started"
https://support.industry.siemens.com/cs/ww/en/view/109740350

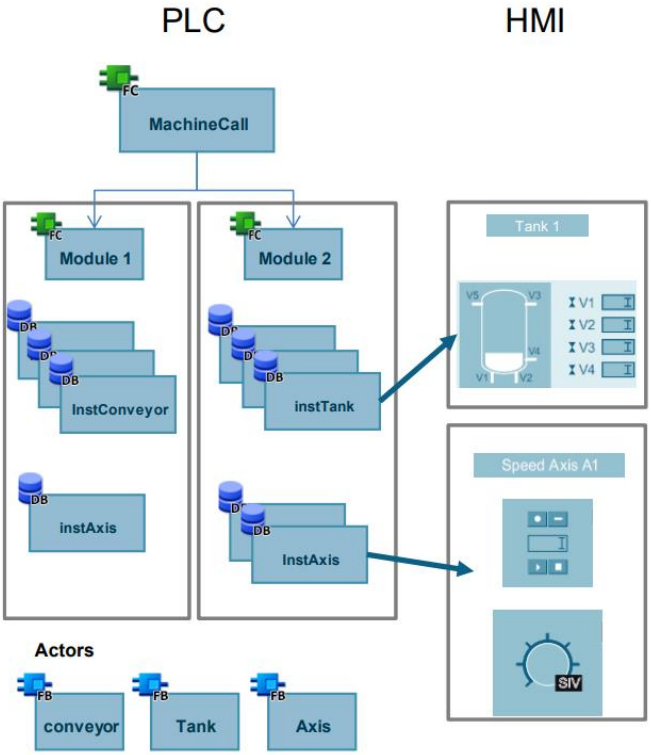
Manual SiVArc
https://support.industry.siemens.com/cs/ww/en/view/109755214

SITRAIN course: SIMATIC Visualization Architect, automatic HMI generating
https://support.industry.siemens.com/cs/ww/en/view/109758628

6.2 How automatic generation works

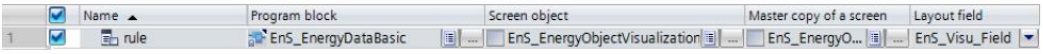
Standardized blocks, e.g. for engine control, are called up in your user program. Using so-called SiVArc rules, you link the called blocks with visualization elements (text fields, IO fields, image blocks, etc.).

Figure 6-1: Example call hierarchy in the control system



SiVArc uses the rules to generate the specified visualization element on a copy of an image template for each call of the specified block.

Figure 6-2: SiVArc rule



Note You also have the option of restricting or blocking the execution of the rule in the controller.

6.3 Controlling the HMI generator

You have the following options for controlling the HMI generator:

- Disable SiVArc generation for specific calls
- Sort SiVArc generation by function or plant location
- Add further properties to SiVArc generation

In SiVArc rules, you can use the following information from the controller, among other things:

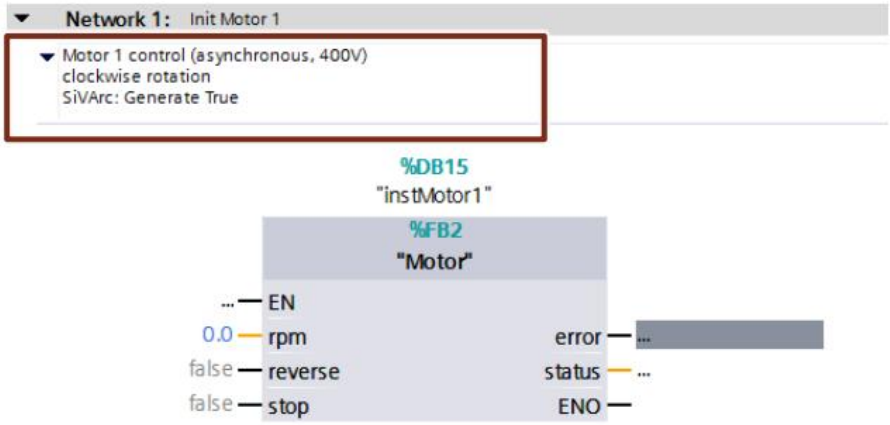
- Network comments
- SiVArc variable

This gives you the option of providing additional information for SiVArc in the controller, which you can use as a condition in the SiVArc rules.

6.3.1 Using network comments for control

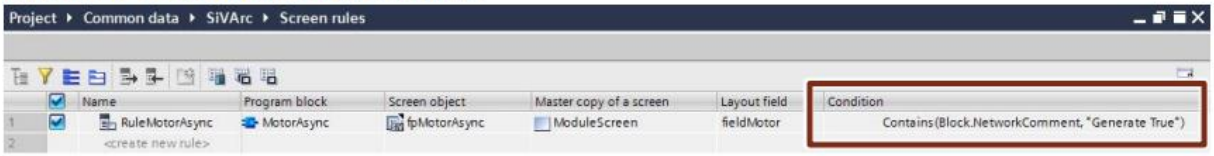
In the network comments, you can add information on controlling the generator that the SiVArc rules search for during generation:

Figure 6-3: Network commentary with SiVArc extension



In the rule editor, use the "Contains" function in the "Condition" column and search for the information, e.g. "Contains(Block.NetworkComment, "String")".

Figure 6-4: Use Network Comment in Rule



With this function you have a free design framework to restrict the execution of rules or to execute a rule only for some networks.

Recommendation

Clearly mark the information for SiVArc generation in the network comments, e.g. "SiVArc: Generate True".

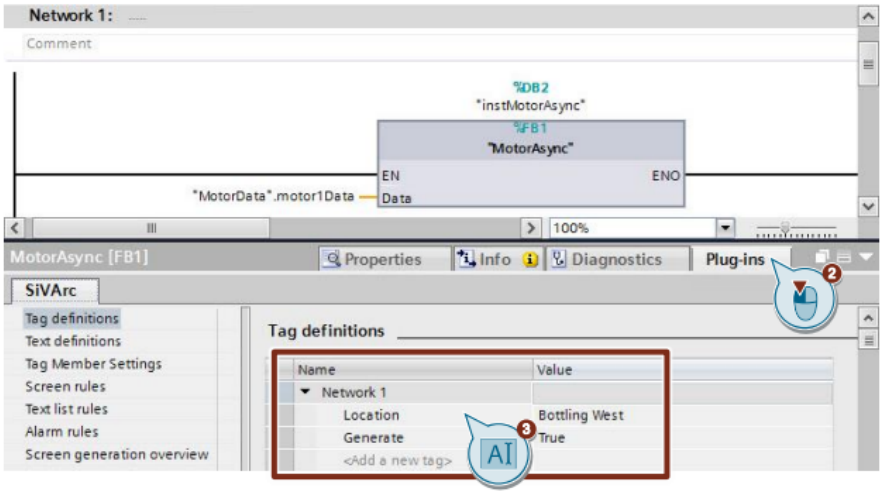
6.3.2 Using SiVArc variables for control

You can define special SiVArc variables for each network in the device and use these variables in the SiVArc rules.

To create SiVArc tags, proceed as follows:

4. Open the block.
5. Switch to the "Plug-Ins" tab in the Inspector window.
6. For each network, enter the name of the variable in the Name column and a value as a string in the "Value" column.

Figure 6-5: Creating SiVArc Variables

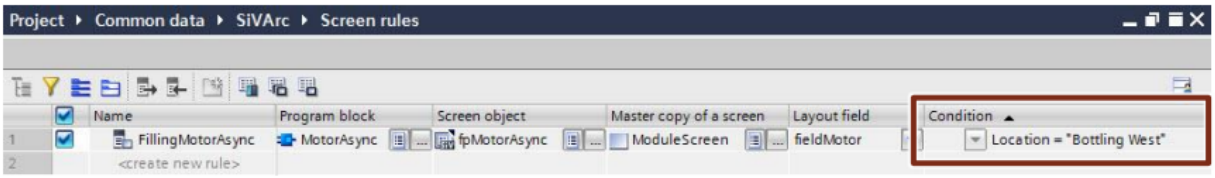


In the rule editor, you can query the SiVArc variables with the expression "variable name = "value"" and thereby influence the exercise of the rule.

In the rule editor, enter the SiVArc variable in the "Condition" column, e.g. "Location = "Bottling West"".

In this example, the rule is only executed if the SiVArc variable "Location" has been set to the value "Bottling West" in the network to be generated.

Figure 6-6: Use SiVArc variable in rule



Recommendation

Use common variable names to control the generator to simplify rule creation. The following table shows an example of SiVArc variables:

Table 8: Example for SiVArc variables

Name	Value	Meaning
Generate	true, false	SiVArc generates elements for this network.
Location	Bottling West, Mixing etc.	You can automatically assign elements to an HMI image using this function.

Note You must define the effects listed in the "Meaning" column yourself using the conditions in the rule editor. The SiVArc variables themselves have no influence on the generator.

6.4 Additional recommendations

Use only characters supported by WinCC

Only use characters that are supported by WinCC for the designation of variables.

When generating the images, SiVArc accesses the identifiers of data blocks, variables or network comments. SiVArc removes all characters that are not supported by WinCC.
This leads to inconsistencies in the project.

Characters not supported:

- %,@, ?, ", /, \, <, >, ., :

Calling the blocks with programming language FBD

So that you can control the generator via network comments or SiVArc variables, call the blocks from which visualization elements are to be generated in the programming language FBD.

Note This recommendation applies up to and including TIA Portal V15.

As of TIA Portal V15.1, SCL modules can also be used.

“

Summary

This chapter is the only chapter solely dedicated to the HMI graphic panel. In Ch. 7, we had a short tutorial involving getting the Siemens’ HMI attached to the S7-1200. That was an introduction to the HMI panel and useful for encouraging students to use the panels instead of wiring to buttons and lights. This chapter expands on that first experience in that both Siemens and A-B are discussed as well as types of product.

The basic panels for both manufactures are introduced and explored. Buttons as well as other devices are built. Some examples of how to use various graphics are included as well. The chapter ends with a discussion of graphics standards and a common problem that I commonly refer as the ‘three-fer’ button. The chapter is not meant to be an exhaustive study of HMI panels but as a starting point for students needing to learn some graphics before launching their careers.

While this chapter begins the broad development of HMI panels, the design of panel interfaces and screen interfaces continue in subsequent chapters, especially the chapter on motion and the chapter on pid control.

Lab 15.1 Revisit - The Cash Register

The basic lab is copied from Chapter 7 as follows:

Design a simple cash register similar to one found at McDonald's or Burger King. To do this, determine a menu of five or six items from the restaurant. Also, include a Total button or a clear button or possibly both. Also, include a means for backing out of a mistake without starting over from zero. Display the cost of the total order in the PLC at an address in the data table. Use floating point math and you are encouraged to do so.

For example:

Whopper Combo	**Whopper**	**Cancel Last**
Whopper Dbl Combo	**Fries**	**New Order**
Whopper Jr Combo	**Drink**	**Total/Tax/ Optional**

Fig. 15-51

Find the approximate prices from a McDonald's or Burger King for the items chosen. When an item is entered, its count is incremented automatically by one. If a button is entered multiple times, the count is incremented to display the total count. If a mistake is made, the attendant must be able to back up at least one entry and erase the last item or decrement that item by one.

Hints to the base lab:

Notice that counters may be referenced as either Count Up or Count Down. If the count is counting up, the count is incremented in rung 0000. If the count is counted down, the count is decremented in rung 0001. Individual inputs are used to increment each product choice. However, to decrement the count, a separate button labeled "Cancel Last" is used. This button must remember the last product chosen and decrement that item. Use the logic in chapter 7 "Relay Instructions" to remember when a button was pushed.

Use the Count Up/Count Down logic for holding active counts for the various items in the cash register.

Make the following changes for the application:

1. Display the total price for the order on the screen. Use Floating Point numbers where possible. Display totals in $xx.xx format.

2. Add a second screen to allow the manager to change base prices for each item. Do not include a password to move from screen to screen.

3. Include a button to add 6.25% tax if not "To Go" for the order.

4. Include a 'live' count of the number of each item ordered.

5. Create means for going from Screen 1 to Screen 2.

6. Screens should resemble the following for **Lab 15.1**:

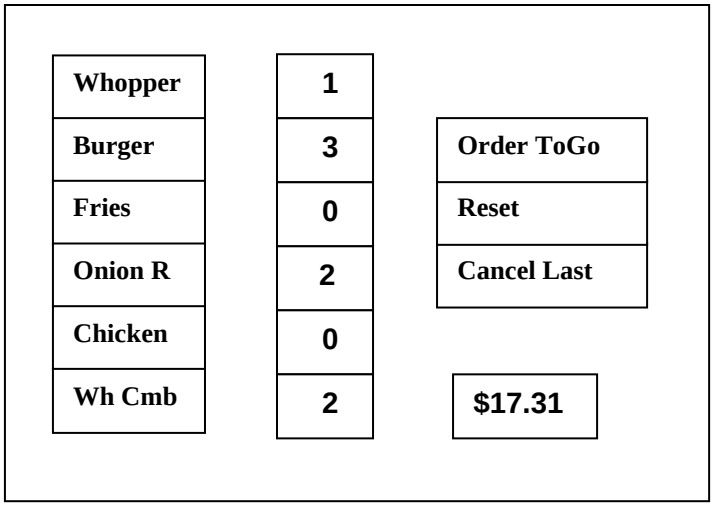


Fig. 15-52

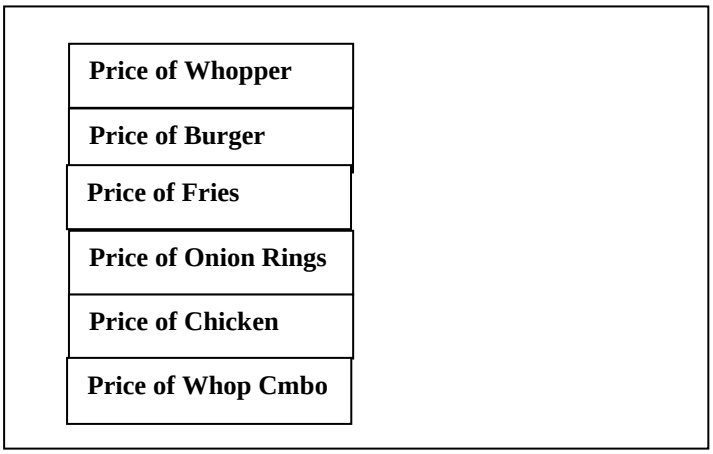


Fig. 15-53

Lab 15.2 Build the Conveyor application as described above for Siemens. Then build the same application for A-B. Compare the two. You will need to write PLC logic to move the elements and increment counts. You do not need to copy the programs included but may write your own programs.

Lab 15.3 Build the graphic for one of the graphics represented in Level 1-3 on pgs 73, 74.
Lab 15.4 Build the graphic for one of the two graphics represented on pg 61.
Lab 15.5 Build the logic and graphic for one of Fig. 15-30, 31, 32 or 38.
Lab 15.6 Build the Siemens HMI program and launch a program from a comfort panel using script. The program can be as simple as 'Hello World'.

Questions

1. How would you build the following pushbuttons in Siemens? Describe:

   Momentary
   Maintained
   Latched
   Multistate
   Interlocked
   Ramp

2. Describe how to demonstrate the flow of a liquid of red color through a series of pipes. Several hand valves are in the path of the flow. How would one describe this graphic with A-B, Siemens?

3. You are assigned the task of describing a ride at an amusement park graphically. The ride is a zip-line. You may place sensors at the beginning and end of the ride. Describe the graphic using A-B and then Siemens controllers and HMI.

4. For the following device, what is the animation property used to move the clock's hand.

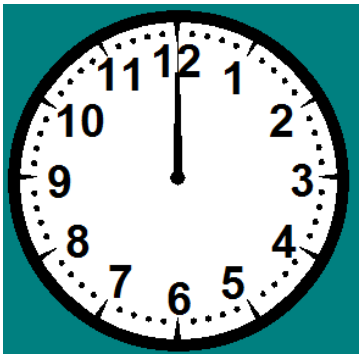


Fig. 15-54

5. Why would one design a PLC and HMI system with OPC and Visual Basic rather than one of the packages described in this chapter?

6. Describe how the programmer would animate the bottle on the conveyor moving right to left for Allen-Bradley, for Siemens.

7. Describe how A-B and Siemens shows each bottle being entered into the case of bottles.

8. Using either A-B or Siemens ladder logic, write a cash register program that uses an accompanying HMI program with only three items [hamburger, fries, drink]. Include logic to calculate the total price ignoring tax. Ignore the 'cancel last' button and combo logic as well.

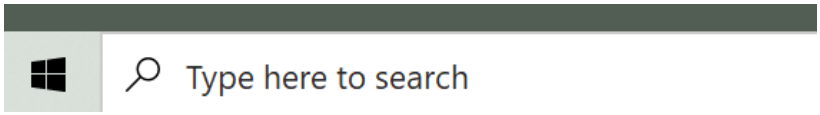
9. Using A-B or Siemens ladder logic, write a program that could be used to show the movement of a bottle across a conveyor belt and populate a case. Use a case of 12. Use a start and stop button to start the operation. Use a reset button to set the bottle counter in the case back to 0 and allow another operation to fill the case with bottles.

10. If you were moving a program from real pushbuttons to an HMI, what one thing must you do?

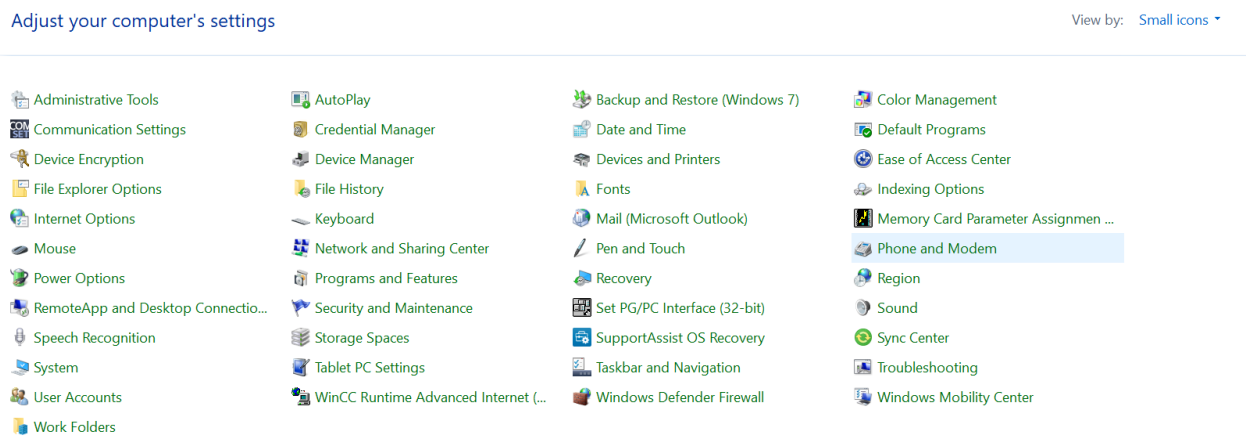
Helps for Getting Going with A-B or Siemens

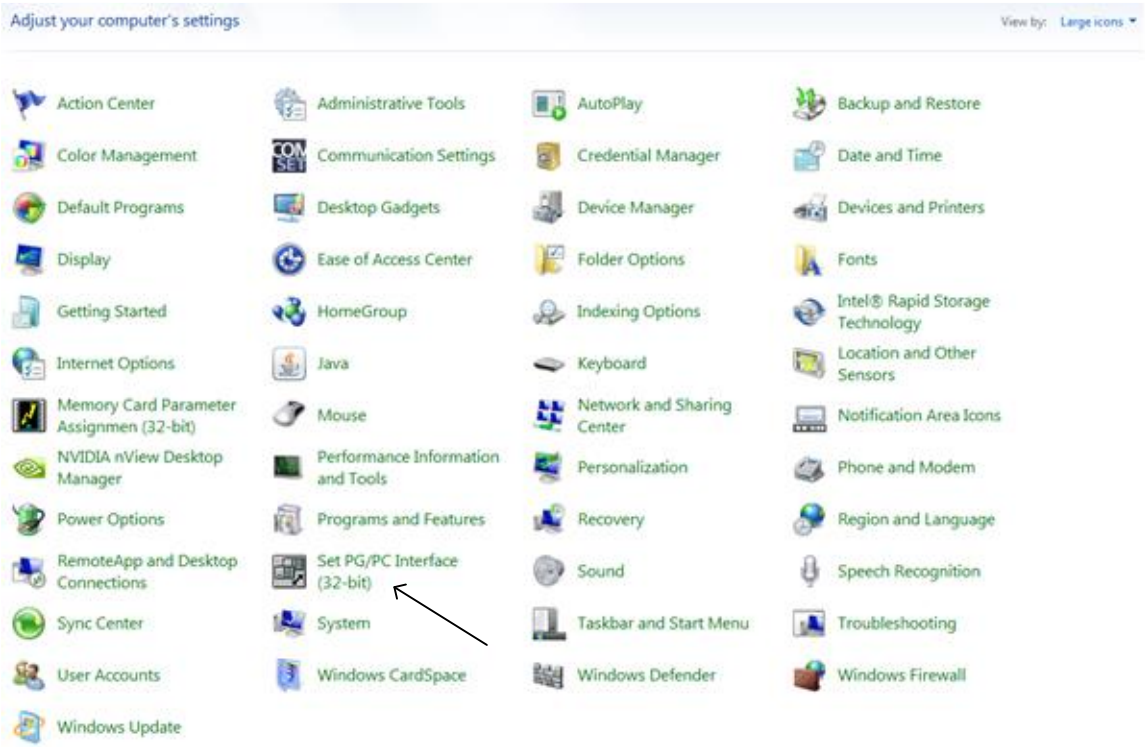
The following must be set up to get the Siemens program to run properly with the HMI simulate mode:

From the area in the lower left corner of the screen, type: control panel



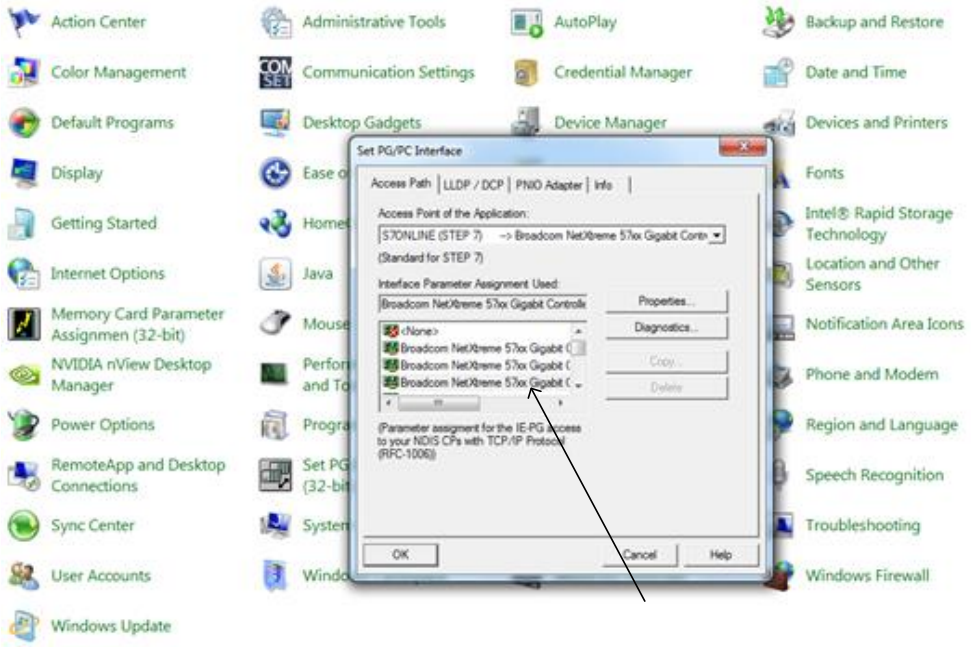
With the Control Panel displayed, look in the upper right and toggle to ‘large icons’:





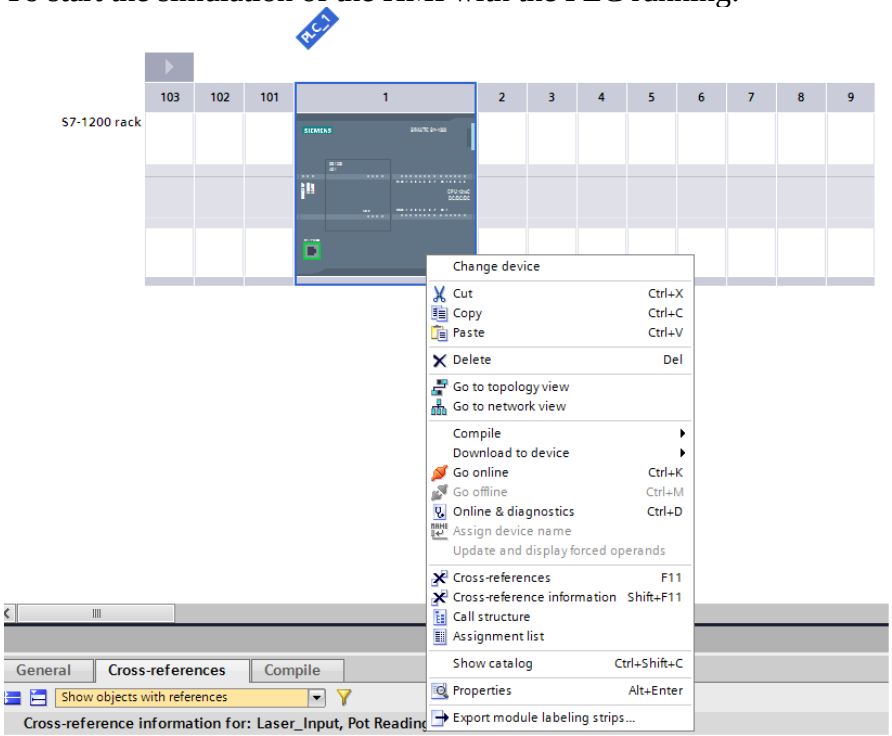
Click on the SetPG/PC Interface box above:

Choose the third of the Broadcom choices. Click OK.



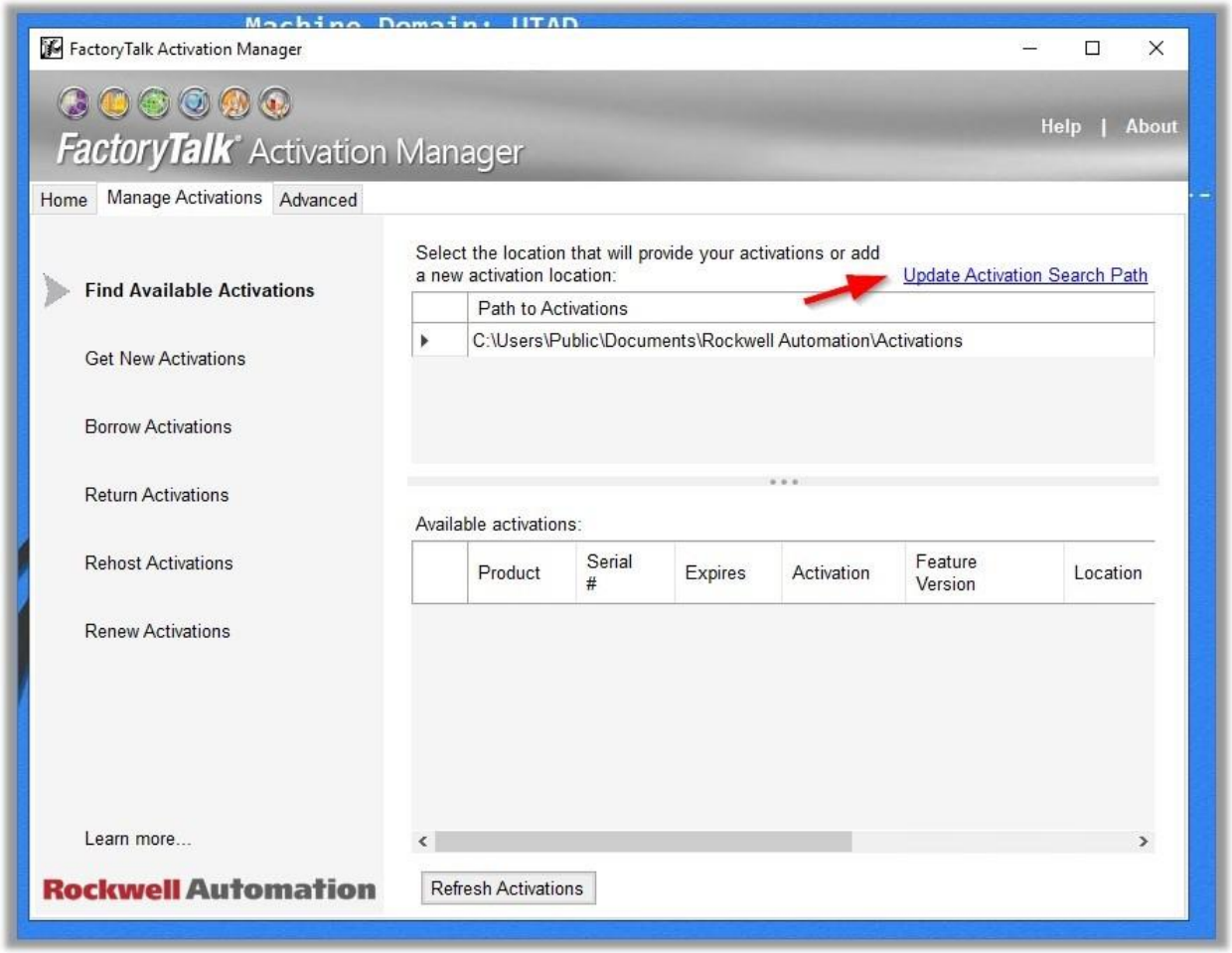
This allows the Siemens program to run the HMI program in simulate mode. Then download the program to the PLC. Do not download the HMI program since we do not have the HMI to download to.

To start the simulation of the HMI with the PLC running:

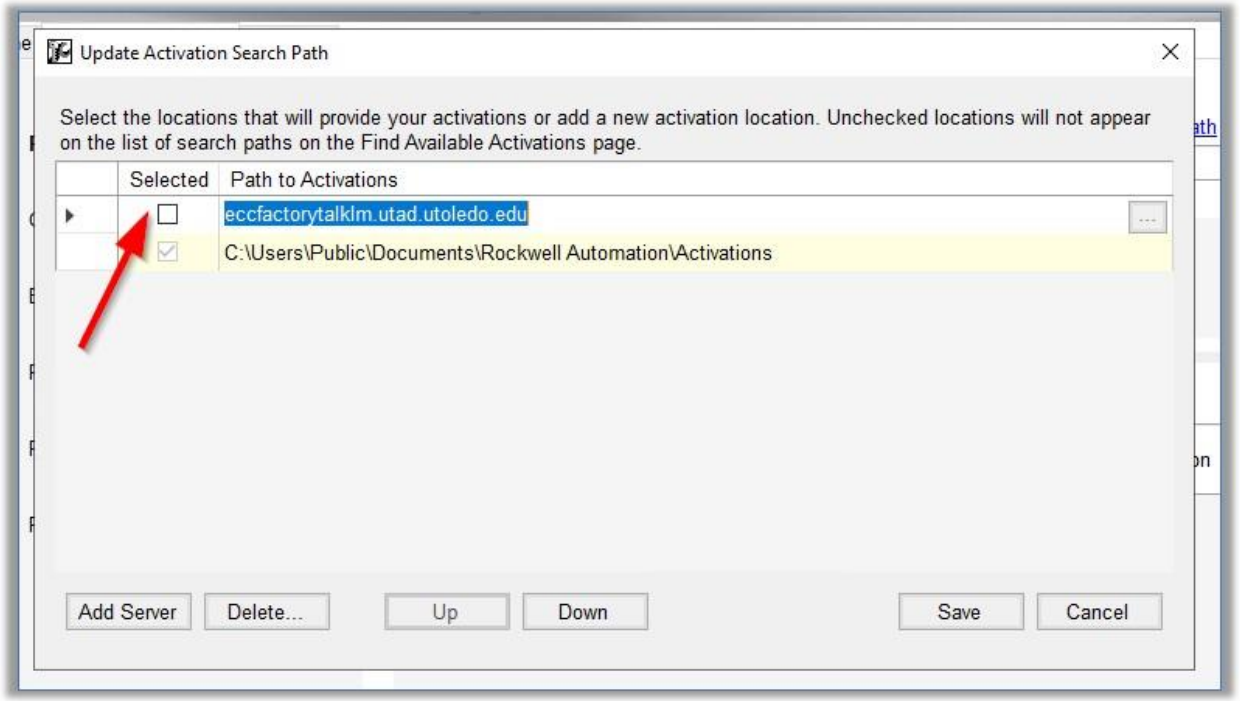


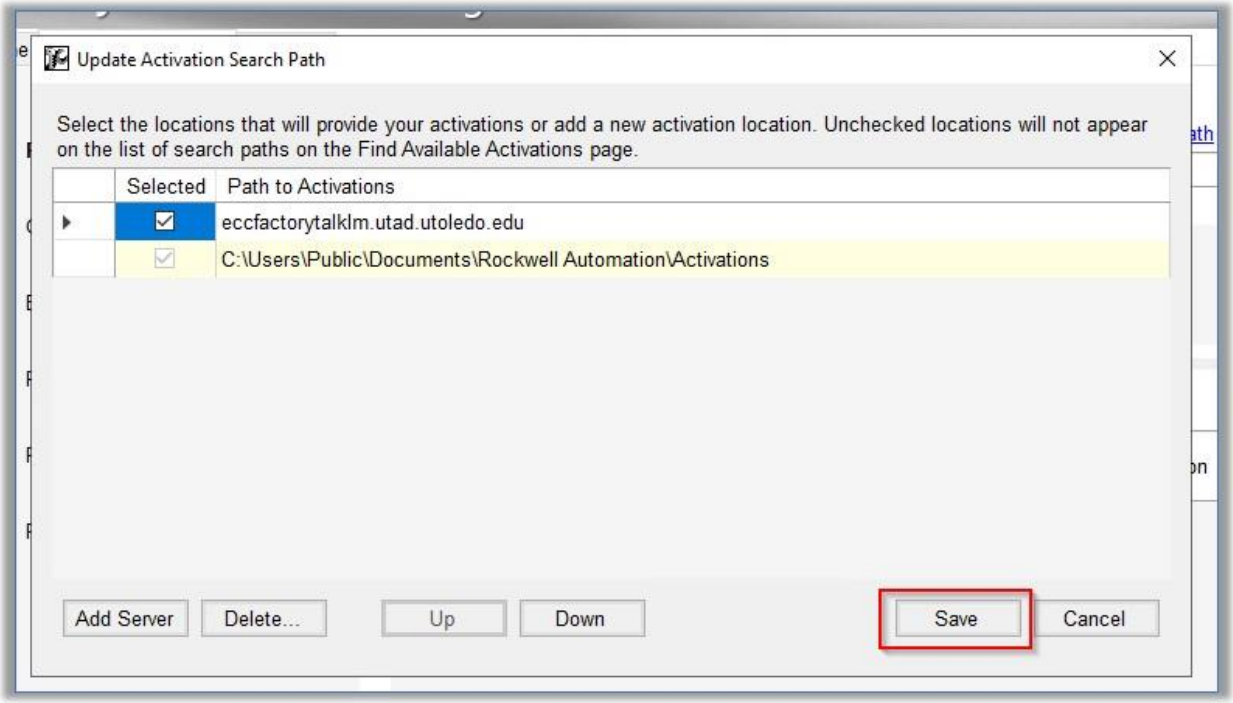
Then click on the HMI's Screen, Screen_1. Notice the Start Simulation button turn blue. It now allows the student to run the HMI via simulation mode from the screen of the pc.

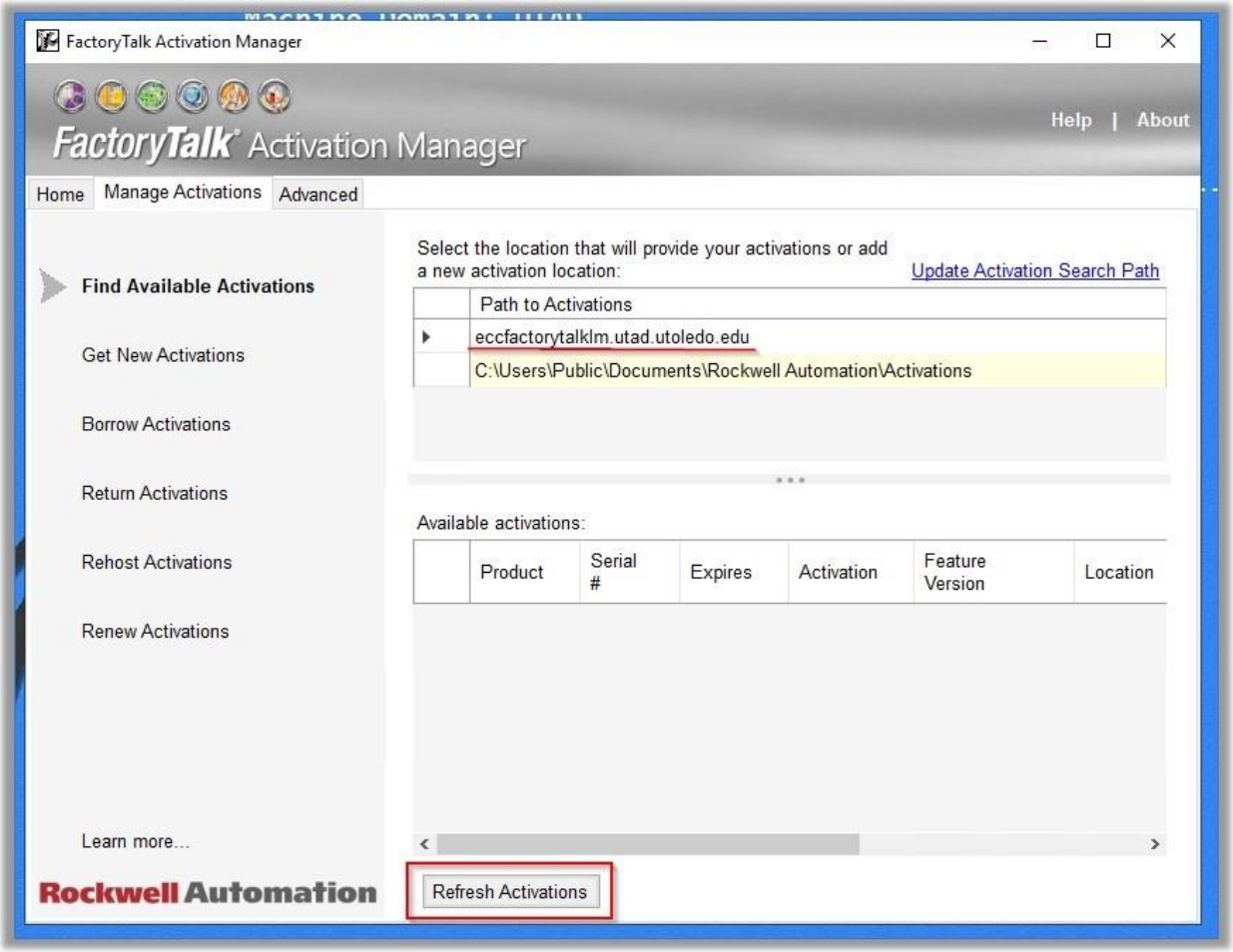
To activate Rockwell software, open the Factory Talk Activation Manger. See the figure below. Click on the “Update Activation Search Path” link on the upper right:

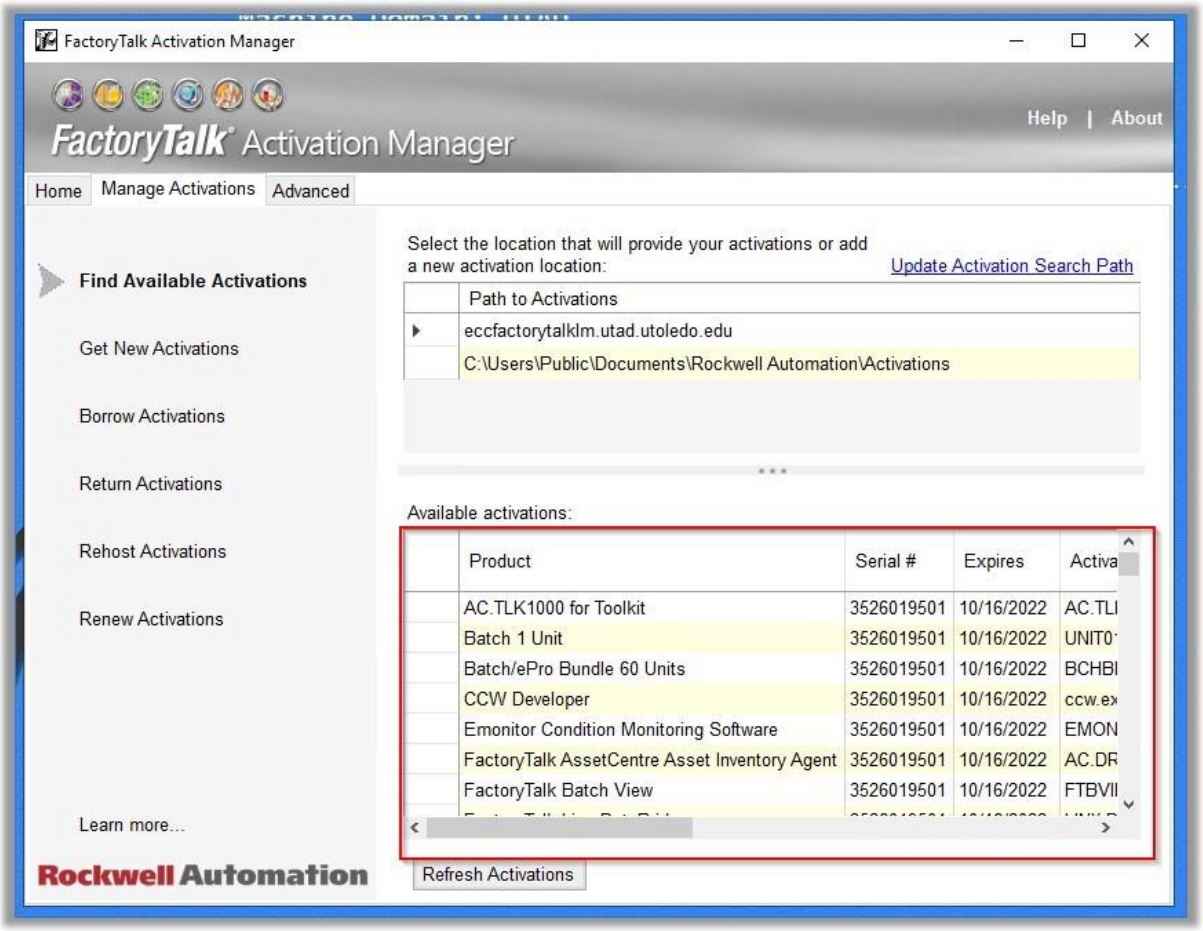


Follow instructions below:




Update Activation Search Path
Select the locations that will provide your activations or add a new activation location. Unchecked locations will not appear on the list of search paths on the Find Available Activations page.
Selected
Path to Activations
eccfactorytalklm.utad.utoledo.edu
C:\Users\Public\Documents\Rockwell Automation\Activations
Add Server
Delete...
Up
Down
Save
Cancel


FactoryTalk Activation Manager
FactoryTalk Activation Manager
Help | About
Home
Manage Activations
Advanced
Find Available Activations
Get New Activations
Borrow Activations
Return Activations
Rehost Activations
Renew Activations
Learn more...
Rockwell Automation
Select the location that will provide your activations or add a new activation location:
Update Activation Search Path
Path to Activations
eccfactorytalklm.utad.utoledo.edu
C:\Users\Public\Documents\Rockwell Automation\Activations
Available activations:
Product
Serial #
Expires
Activation
Feature Version
Location
Refresh Activations



You should be activated again!

This work is licensed under a Creative Commons Attribution 4.0 International License.