TouchPAD

TPD/VPD Series HMI Device

User Manual Version 1.0.9

ICP DAS Co., Ltd.

Warning

ICP DAS assumes no liability for any damage resulting from the use of this product. ICP DAS reserves the right to change this manual at any time without notice. The information furnished by ICP DAS is believed to be accurate and reliable. However, no responsibility is assumed by ICP DAS for its use, nor for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only may be registered trademarks of their respective companies.

Support

ICP DAS takes your problem as ours.

If you have any problem, please feel free to contact us.
You can count on us for quick response.

Email: service@icpdas.com

Tel: 886-3-5973336

Also, the FTP site of ICP DAS has contents about TouchPAD which you may be interested in. We believe that those contents may be helpful to your work.

FTP: ftp://ftp.icpdas.com/pub/cd/touchpad/

Preface

Thank you for buying TPD/VPD Series HMI Devices, TouchPADs, which are made by ICP DAS Co., Ltd. We suggest you read through this user manual before you set up these devices and develop their programs.

Purpose

- This manual shows how to use TouchPADs and develop programs.
- This manual mainly contains the following parts:
  - Introduction: basic understandings of TouchPADs.
  - Hardware: specifications, dimensions, and installations.
  - Software: mainly how to build a project and HMIWorks introductions.

Personnel

This manual is fit for following personnel:

- End Users
- Engineers
- Technicians

Table of Contents

Preface 3
1. Introduction 7
   1.1. Advanced Features 7
   1.2. Applications of TouchPAD 8
   1.3. All Kinds of Situations 10
2. General Specifications 13
   2.1. TPD-28x Series 14
   2.2. TPD-28xU Series 16
   2.3. TPD-43x Series 17
   2.4. VPD-13x Series 18
3. Hardware 21
   3.1. Hardware Appearance 21
   3.2. Dimensions 24
   3.3. Installation and Wiring 30
4. Setup Devices and Connect to I/O 36
   4.1. Preparation 36
   4.2. Setup Devices 37
      4.2.1. Setup TPD-280 37
      4.2.2. Setup TPD-283 39
      4.2.3. Setup Other Devices in TouchPAD Series 47
   4.3. Connect to I/O Modules 53
5. Development Software, HMIWorks 61
   5.1. The Construction of HMIWorks 61
   5.2. The Options of TouchPAD 64
   5.3. Ladder Designer 65
      5.3.1. Getting Started 66
      5.3.2. Introduction to Ladder Designer 67
      5.3.3. Operations of Ladder Designer 72
      5.3.4. User-Defined Function Block 86
      5.3.5. Associate Tags with Tools 89
      5.3.6. User-Defined I/O Modules 91
   5.4. Frames and Tools 98
      5.4.1. Properties of a Frame 99
      5.4.2. Drawing a Rectangle 102

5.4.3. Drawing an Ellipse 105
5.4.4. Drawing a Text 107
5.4.5. Loading a Picture 110
5.4.6. Drawing a Line 113
5.4.7. Using a TextPushButton 115
5.4.8. Using a Slider 120
5.4.9. Using a BitButton 124
5.4.10. Using a HotSpot 128
5.4.11. Using a CheckBox 131
5.4.12. Using a Label 135
5.4.13. Using a Timer 140
5.4.14. Using a PaintBox 142
5.4.15. Using an ObjectList 145
5.5. Menus 150
5.5.1. Starting Design, File Menu 151
5.5.2. Cascading and Grouping, Arrange Menu 152
5.5.3. Rotating and Flipping, Edit Menu 154
5.5.4. Gridding and Viewing, View Menu 157
5.5.5. Frame Managing and Aligning, Layout Menu 159
5.5.6. Build and Download to Run, Run Menu 163
5.5.7. Library Management, Popup Menu 164
6. Make a Simple Project 168
6.1. Your First Project Using Standard C 168
6.2. Your First Project Using Ladder 171
6.3. Integrate TPD-280 Series with I/O modules 176
6.4. Integrate TPD-283 Series with I/O modules 179
7. Advanced Programming in C 183
7.1. Add a New File to Project 183
7.2. Update Properties in Run Time 184
7.2.1. FillColor and Text of a TextPushButton 184
7.2.2. Percentage of a Slider 186
7.2.3. Selected of a CheckBox 187
7.2.4. Font, Text and TextColor of a Label 188
Appendix 192
A. FAQ 192
A.1. What to do if screen flashes? 192
A.2. How to have higher-resolution Picture? 193
A.3. How does a TouchPAD control I/O? 194

A.4. How to change Font of Text? 195
A.5. How to represent decimals for Ladder Designer? 196
A.6. How to remove the startup beep of TPD-283? 197
A.7. How to store data in the flash? 198

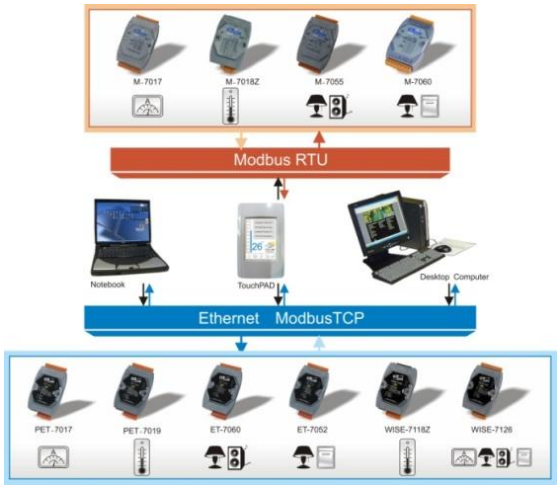
1. Introduction

Our solution for HMI (Human Machine Interface) is composed of GUI (Graphical User Interface) based touch screens and an integrated software development package. ICP DAS hears the voices of our customers and is dedicated to providing a series of solutions particularly for intelligent building, equipment monitoring, factory automation and automatic controls. Its development software, HMIWorks, provides plenty of widgets and a variety of templates. Combined with the high resolution color touch screen of the TouchPAD series, a GUI can be realized with your own unique fashion and style. Development is no longer difficult and project accomplishment is within reach.

ICP DAS provides two types of touch HMI devices, the TPD series and the VPD series. The TPD series is designed for home/building automation applications and the VPD series is designed for factory/machine automation applications. Both have many common features, such as a high-resolution touch screen, RTC, and a variety of communication interfaces, including RS-232/RS-485, Ethernet, USB. However, each still has its own specific features for its respective target applications. For the TPD series, you can use an external wall box to help you smoothly blend the TPD series device into your decoration. For the VPD series, the rubber keypad, IP-65 waterproof front panel and DIN-Rail/panel mounting are designed for harsh environment, and are especially suitable for factories.

1.1. Advanced Features

- Excellent C/P ratio (cost/performance)
- Workable under tough environments, operating temperature: -20℃~70℃
- High Color resolution touch screen
- PoE, Power over Ethernet (TPD-283)
- RS-485 network (TPD-280/TPD-280U/TPD-430/VPD-130)



- WYSIWYG (What You See Is What You Get) GUI design
- Complete and powerful development tool, easy integration with touch HMI devices, quick design for a variety of applications
- Supports the popular C programming language
- Ladder logic design

1.2. Applications of TouchPAD

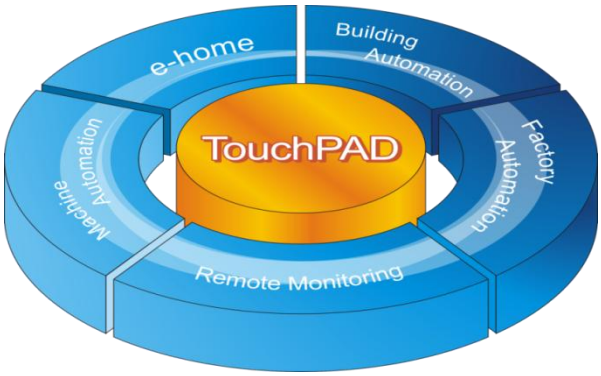
Interactions between human and machine are getting more and more important since automation control has emerged. From systematic surveillance of equipments, status monitoring of house appliances or even measurements of temperature and humidity, HMI devices play an indispensable role in passing information. In the early time, HMI devices consist of lights, meters, 7-segment display. For now, LED and LCD are prevalently used and ICP DAS releases TouchPAD as a state-of-the-art solution.

In addition to GUI and touch LCD, the solution of ICP DAS provides development software package, HMIWorks. HMIWorks satisfies most of the requirements with the WYSIWYG (What You See Is What You Get) design environment. Besides, It has plenty of widgets and varieties of templates, and so it's easy to present customers with professional interface with modern styles. Moreover, HMIWorks supports C language. It makes easy timing control and logic design and in turns makes TouchPAD more powerful. Through standard communication protocols and SCADA (Supervisory Control and Data Acquisition) software, it is directly inoculated with background

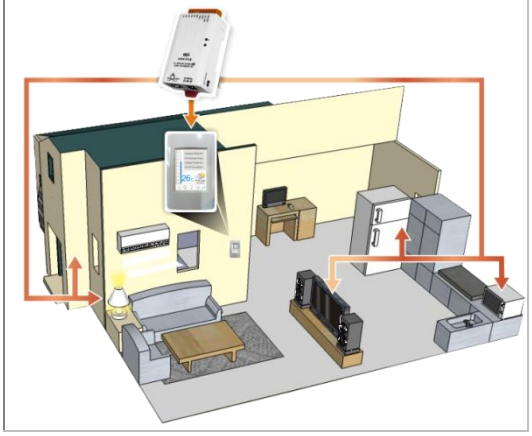
software and attains the best effect of completely integration.

Below are figures of the application fields of TouchPAD, intelligent building and classroom automation. In these examples, TouchPADs are used to control lights, curtains, air conditioners, stereos, projectors, projector screens, and to monitor temperature, humidity, and weather conditions.

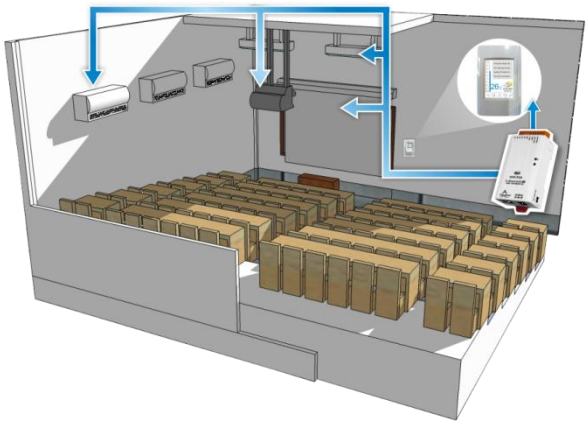
Application Fields of TouchPAD



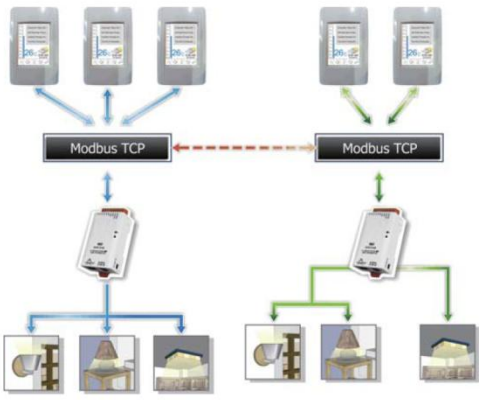
Intelligent Building Example



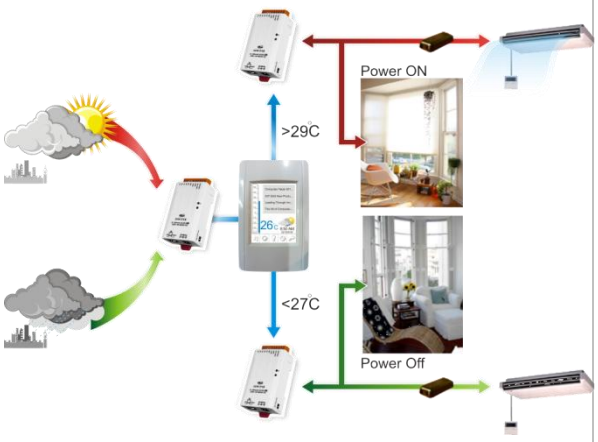
Classroom Automation Example

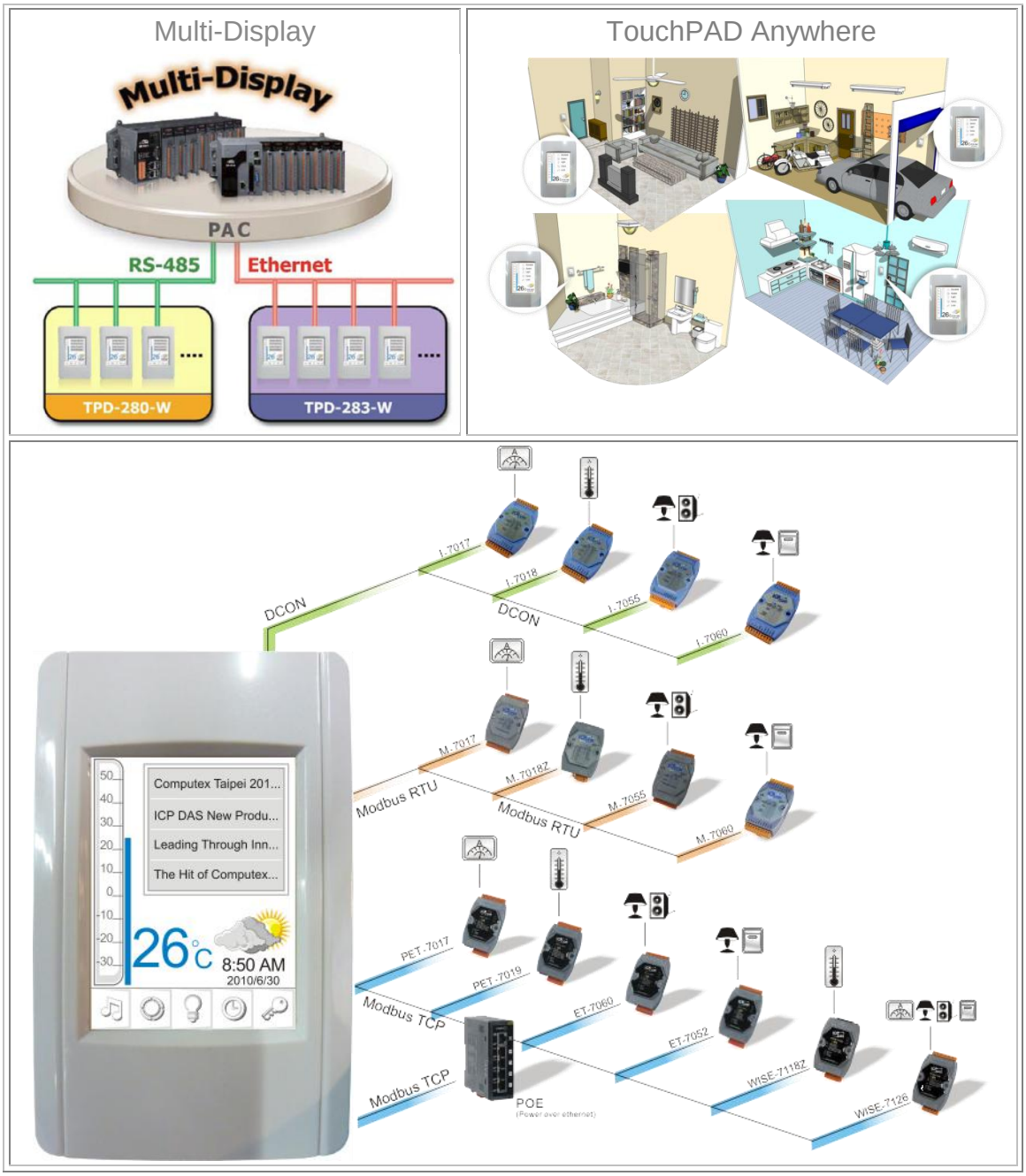


Single-way, Two-way, Multi-way Switches Example



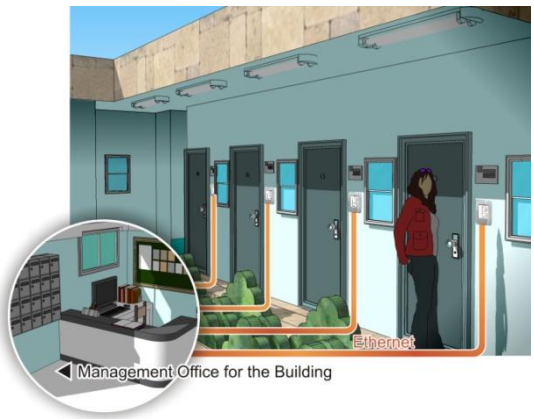
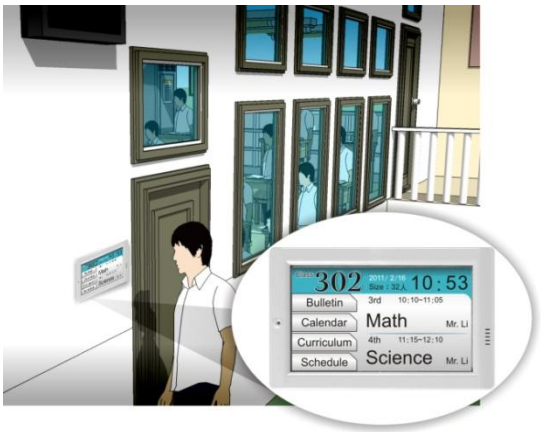
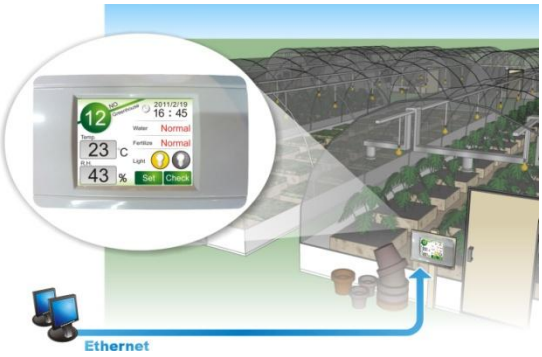
Temperature Control Example

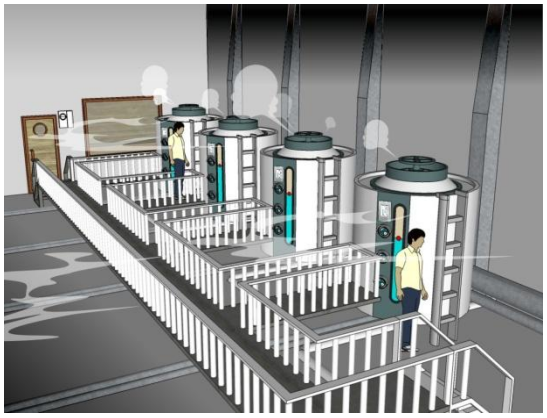
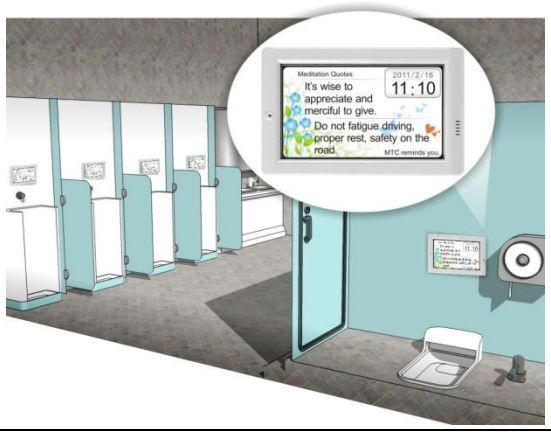




1.3. All Kinds of Situations

TouchPAD can be applied in all kinds of situations you even have imagined. In these situations, TouchPAD is used as a window for signal transmissions and arranges incoming messages.

In the hotel, on the wall 	In the apartment building, beside the door 
In the exhibition hall, beside the exhibits 	In the school, in the classroom 
In the office, outside the meeting room 	In the farm, in the front door of the greenhouse 

In the hypermarket, on the shelves 	In the factory, on the steam chambers 
In the toilet 	And more…

2. General Specifications

The format of the model name for the TPD series is as follows:

T P D - **XX** **X(X)** **-** **XX**

- **XX** — touch screen size
  - 28: 2.8 inch
  - 43: 4.3 inch
- **X(X)** — communication interface
  - For 2.8 inch:
    - 0: RS-485
    - 0U: RS-485 + USB
    - 3: Ethernet
  - For 4.3 inch and above:
    - 0: RS-485 + USB
- **XX** — EU: for European 86 x 86 mm Outlet Box

Model Name	Memory Expansion	Image Storage Capacity	Communication Interface	LCD	RTC	USB 1.1 Client	Outlet Box	Power Input
TPD-280	-	1	RS-485 (including Self-Tuner)	2.8" TFT (Resolution 240 x 320 x 16)	-	-	-	+10 ~ 30 V_{DC}
TPD-280U	16 MB SDRAM/ 8 MB Flash	54			Yes	Yes	-	
TPD-283	-	1	Ethernet (10/100 Mbps)		-	-	-	PoE (IEEE 802.3af, Class 1)
TPD-430	16 MB SDRAM/ 8 MB Flash	32	RS-485 (including Self-Tuner)	4.3" TFT (Resolution 480 x 272 x 16)	Yes	Yes	Suitable for the Outlet Box in United States	+10 ~ 30 V_{DC}
TPD-430-EU		32					Suitable for the European 86 x 86 mm Outlet Box	

* Image Storage Capacity strongly depends on the content and the size of images. The number in the table shows how many images in full screen size can be stored on the device.

The format of the model name for the VPD series is as follows:

V P D - **X** **X** **X(X)**

- **X** — form factor
  - 1: 103 × 103 mm Panel Mount
  - 2: 182 × 158 mm Panel Mount
- **X** — display size
  - 2: 2.8 inch Display
  - 3: 3.5 inch Display
- **X(X)** — Communication interface
  - 0: RS-485 + Rubber Keypad
  - 0N: RS-485

Model Name	Memory Expansion	Image Storage Capacity	Communication Interface	LCD	RTC	USB 1.1 Client	Rubber Keypad	Ingress Protection	Power Input
VPD-130	16 MB SDRAM/ 8 MB Flash	54	RS-232/RS-485 (including Self-Tuner)	3.5" TFT (Resolution 320 x 240 x 16)	Yes	Yes	Yes	Front Panel: IP65	+12 ~ 48 V_{DC}
VPD-130N		54					-		

* Image Storage Capacity strongly depends on the content and the size of images. The number in the table shows how many images in full screen size can be stored on the device.

Note:

Communication interface that is only for run time supports the following protocols:

I. For the case of RS-485, Modbus RTU Master and DCON Protocol

Master (for ICP DAS I-7000 series modules) are supported.
We provide API functions to open com port for sending/receiving strings through RS-485.

II. For the case of Ethernet, Modbus TCP Master is supported.
We provide API functions to sending/receiving strings through TCP.

USB is used for firmware update only.

Presently, the expansion flash is used **only** to store a firmware, not logging data.

2.1. TPD-28x Series

In contrast to the TPD-28xU series, TPD-28x series devices are not USB clients. Besides, TPD-28x series does not have SDRAM/Flash expansion.

Models	TPD-280	TPD-283
Image		
CPU Module		

CPU	32-bit RISC CPU	
Buzzer	Yes	
Rotary Switch (0~9)	Yes	
Communication Interface		
Ethernet	-	RJ-45 x 1, 10/100 Base-TX
Serial Port	RS-485 (including Self-Tuner)	-
MMI (Main Machine Interface)		
LCD	2.8" TFT (Resolution 240 x 320 x 16), defective pixels <= 3	
Backlight Life	20,000 hours	
Brightness	160 cd/m2	
Touch Panel	Yes	
Reset Button	Yes	
Electrical		
Power Input Range	+10 ~ 30 V_{DC}	PoE (Power over Ethernet)
Power Consumption	1.2 W (50 mA @ 24 V_{DC})	IEEE 802.3af, Class 1
Mechanical		
Dimensions (WxLxH)	76mm (W) x 119mm (L) x 31mm (H)	
Installation	Wall Mount	
Environmental		
Operating Temperature	-20 ~ +70 °C	
Storage Temperature	-30 ~ +80 °C	
Ambient Relative Humidity	10 ~ 90% RH, non-condensing	

2.2. TPD-28xU Series

In contrast to the TPD-28x series, the TPD-28xU series devices are USB clients for the purpose of firmware update. This is the meaning of the **U** in the TPD-28x**U** series. Moreover, the TPD-28xU series has SDRAM/Flash expansion.

Models	**TPD-280U**
Image	
CPU Module	
CPU	32-bit RISC CPU
Memory Expansion	16 MB SDRAM / 8 MB Flash
Real Time Clock (RTC)	Yes
Buzzer	Yes
Rotary Switch (0~9)	Yes
Communication Interface	
Serial Port	RS-485 (including Self-Tuner)

USB 1.1 Client	Firmware updates only
MMI (Main Machine Interface)	
LCD	2.8” TFT (Resolution 240 x 320 x 16), defective pixels <= 3
Backlight Life	20,000 hours
Brightness	160 cd/m2
Touch Panel	Yes
Reset Button	Yes
Electrical	
Power Input Range	+10 ~ 30 VDC
Power Consumption	1.2 W (50 mA @ 24 VDC)
Mechanical	
Dimensions (WxLxH)	76mm (W) x 119mm (L) x 31mm (H)
Installation	Wall Mount
Environmental	
Operating Temperature	-20 ~ +70 °C
Storage Temperature	-30 ~ +80 °C
Ambient Relative Humidity	10 ~ 90% RH, non-condensing

2.3. TPD-43x Series

This series is 4.3” touch screen HMI devices.

Models	**TPD-430**	**TPD-430-EU**
Image		
CPU Module		
CPU	32-bit RISC CPU	
Memory Expansion	16 MB SDRAM / 8 MB Flash	

Real Time Clock (RTC)	Yes	
Speaker	Yes	
Rotary Switch (0~9)	Yes	
Communication Interface		
Serial Port	RS-485 (including Self-Tuner)	
USB 1.1 Client	Firmware updates only	
MMI (Main Machine Interface)		
LCD	4.3" TFT(Resolution 480 X 272 X 16), defective pixels <= 3	
Backlight Life	20,000 hours	
Brightness	400 cd/m2	
Touch Panel	Yes	
LED Indicator	Yes	
Reset Button	Yes	
Electrical		
Power Input Range	+10 ~ 30 V_{DC}	
Power Consumption	2.5 W (104 mA @ 24 VDC)	
Mechanical		
Dimensions (WxLxH)	126mm(W) x 82mm(L) x 24mm(H)	126mm(W) x 92mm(L) x 29mm(H)
Installation	Wall Mount (Suitable for the outlet box in United States)	Wall Mount (Suitable for the European 86mm x 86mm outlet box)
Environmental		
Operating Temperature	-20 ~ +70 °C	
Storage Temperature	-30 ~ +80 °C	
Ambient Relative Humidity	10 ~ 90% RH, non-condensing	

2.4. VPD-13x Series

VPD series devices are designed for industrial applications.

Models	**VPD-130**	**VPD-130N**

Image		
CPU Module		
CPU	32-bit RISC CPU	
Memory Expansion	16 MB SDRAM / 8 MB Flash	
Real Time Clock (RTC)	Yes	
Buzzer	Yes	
Rotary Switch (0~9)	Yes	
Communication Interface		
Serial Port	One set of RS-232/ RS-485, including Self-Tuner	
USB 1.1 Client	Firmware updates only	
MMI (Main Machine Interface)		
LCD	3.5” TFT (Resolution 240 x 320 x 16), defective pixels <= 3	
Backlight Life	20,000 hours	
Brightness	270 cd/m2	
Touch Panel	Yes	
LED Indicator	Yes	-
Reset Button	Yes	
Rubber Keypad	5 keys (Programmable)	-
Electrical		
Power Input Range	+12 ~ 48 VDC	
Power Consumption	2 W (83 mA @ 24 VDC)	
Mechanical		
Dimension	103 mm (W) x103 mm (L) x 53mm (H)	
Ingress Protection	Front Panel: IP65	
Installation	DIN-Rail Mount and Panel Mount	
Environmental		
Operating Temperature	-20 ~ +70 °C	
Storage Temperature	-30 ~ +80 °C	

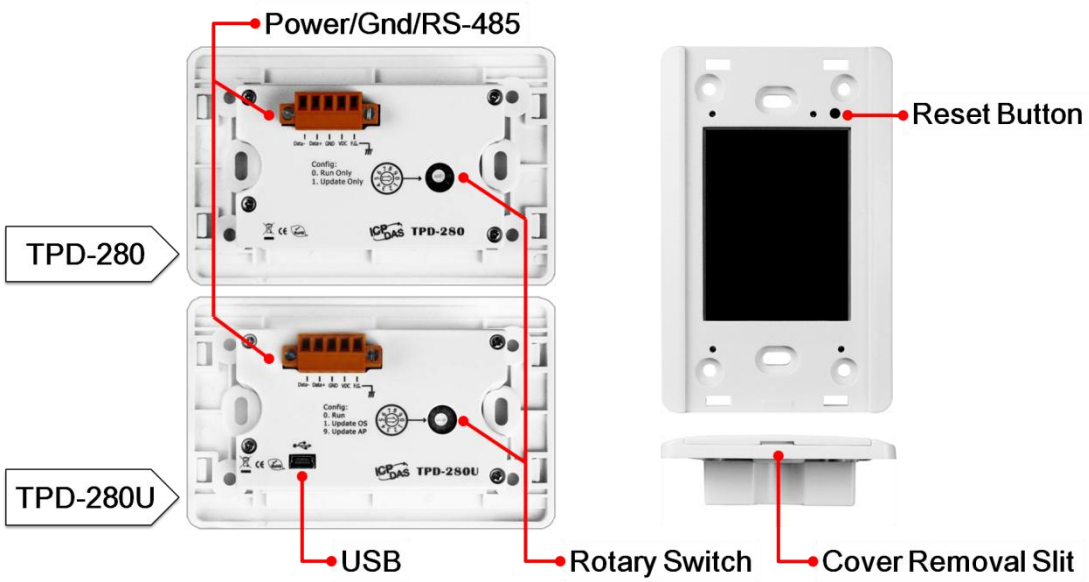
Ambient Relative Humidity	10 ~ 90% RH, non-condensing

3. Hardware

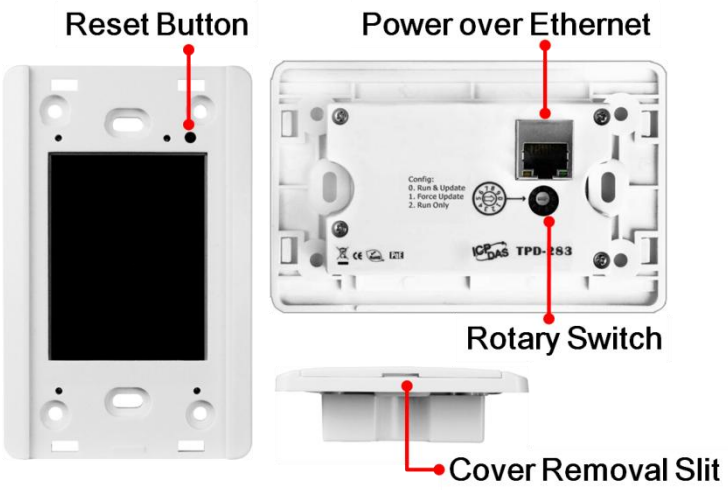
This chapter shows the overviews, dimensions, etc. of TouchPAD.

3.1. Hardware Appearance

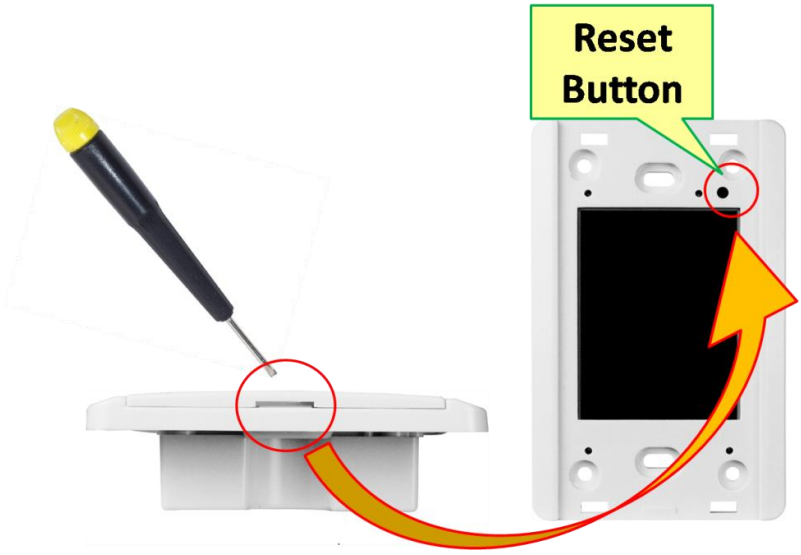
TPD-280/TPD-280U



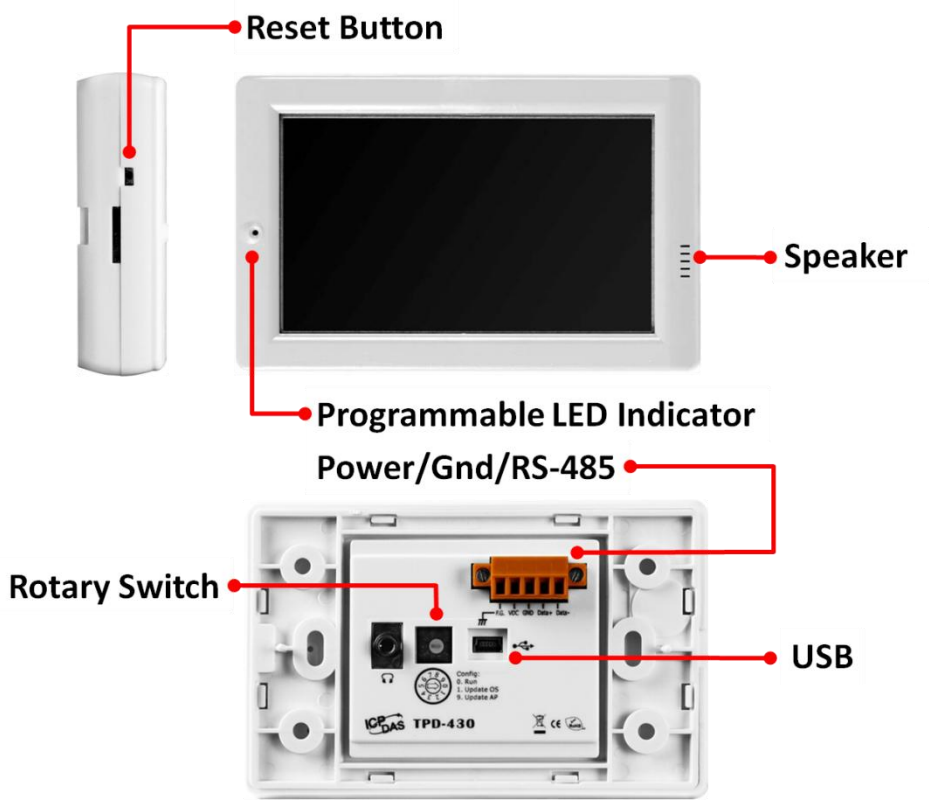
TPD-283



Where is the reset button?



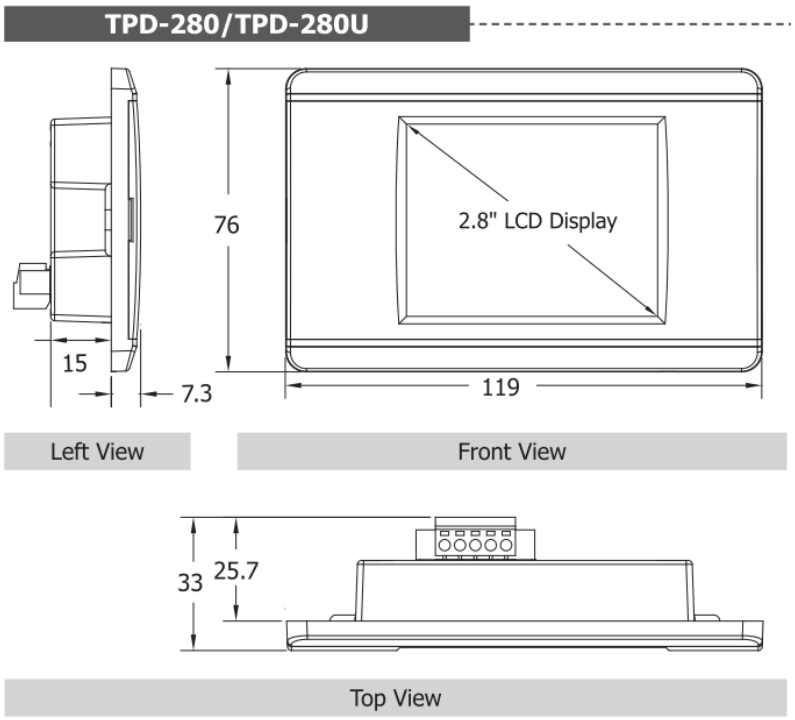
TPD-430/TPD-430-EU

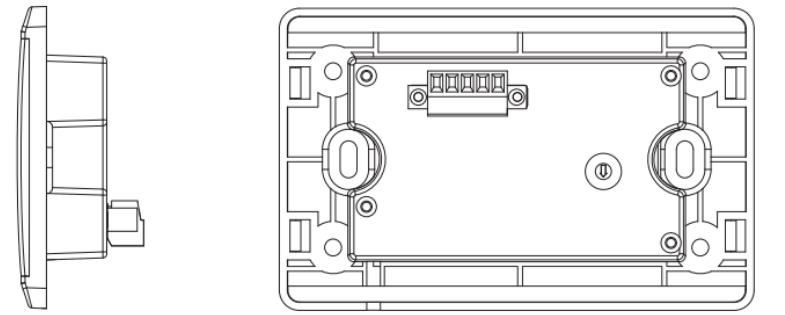


VPD-130/VPD-130N

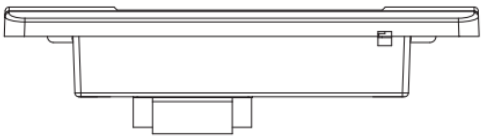
3.2. Dimensions

TPD-280/TPD-280U (unit: mm)



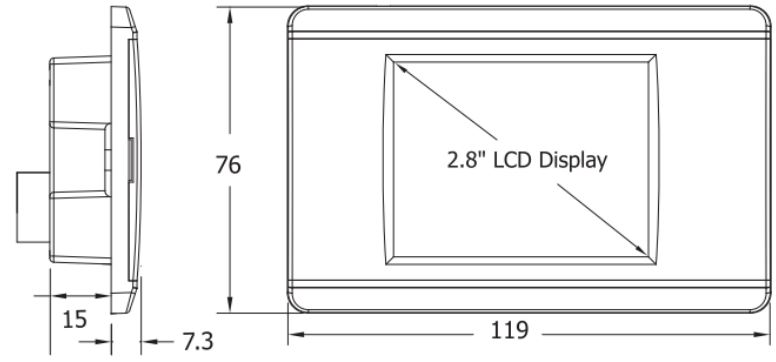


Right View | Rear View

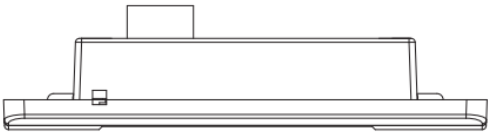


Bottom View

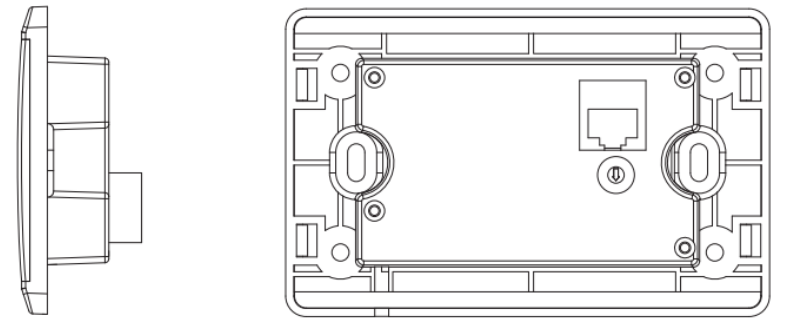
TPD-283 (unit: mm)



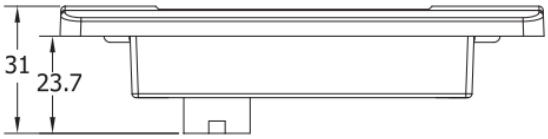
Left View | Front View



Top View

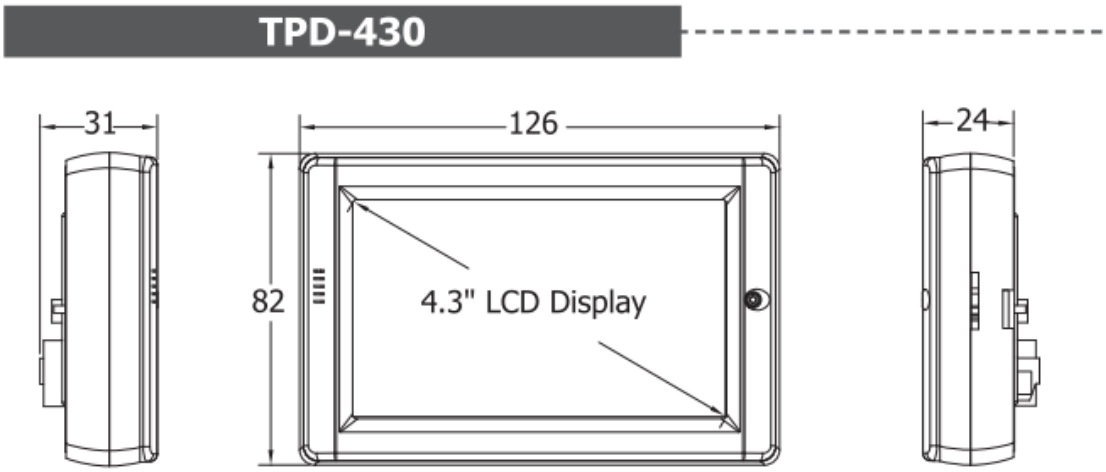


Right View | Rear View

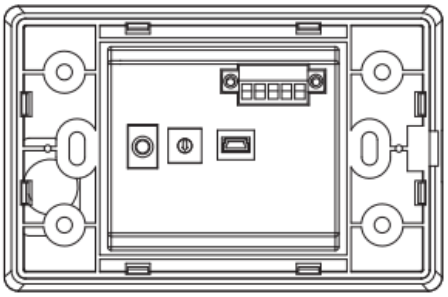


Bottom View

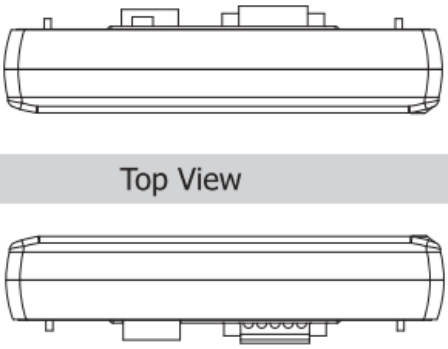
TPD-430/TPD-430-EU (unit: mm)



Left View | Front View | Right View

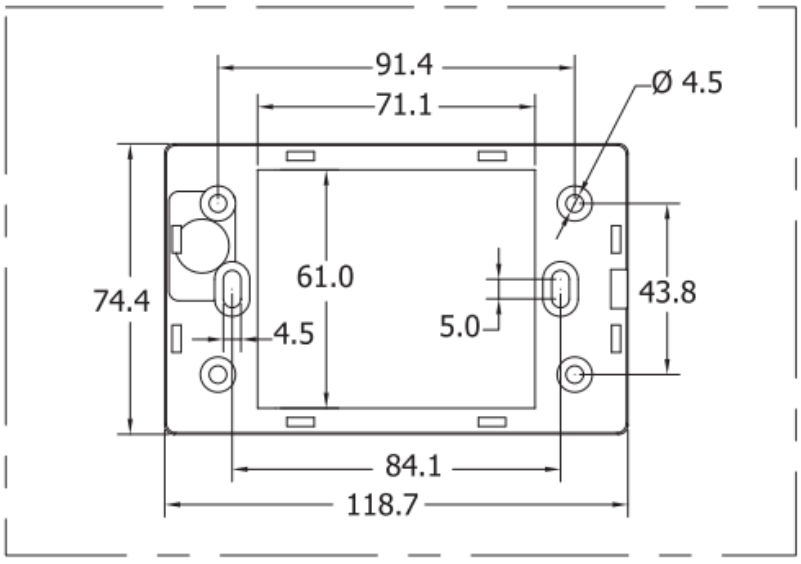


Rear View



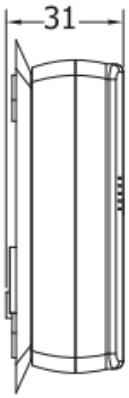
Top View

Bottom View

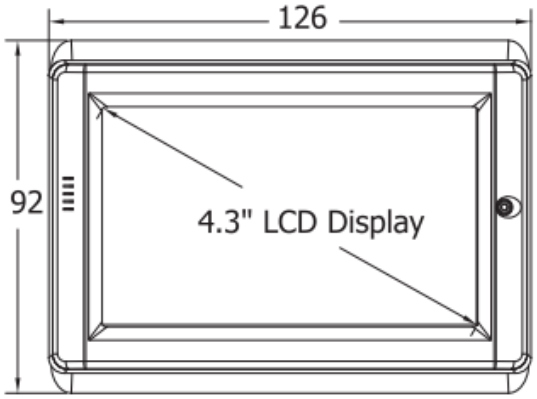


Wall Mounting

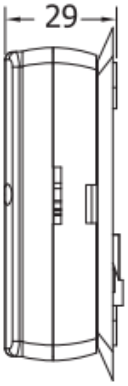
TPD-430-EU



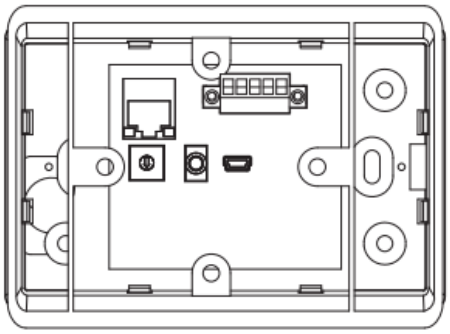
Left View



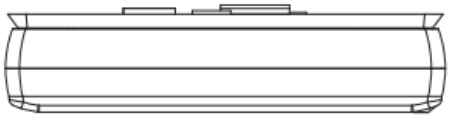
Front View



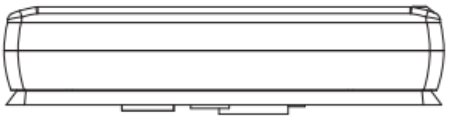
Right View



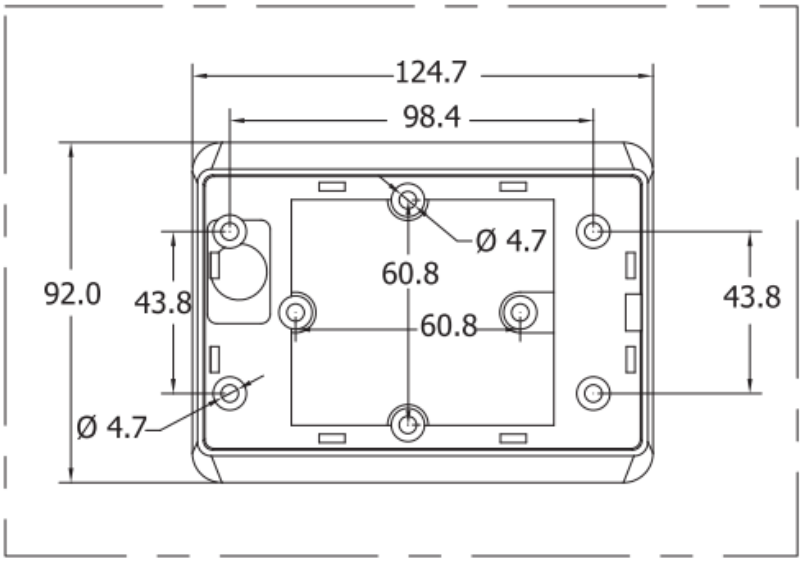
Rear View



Top View



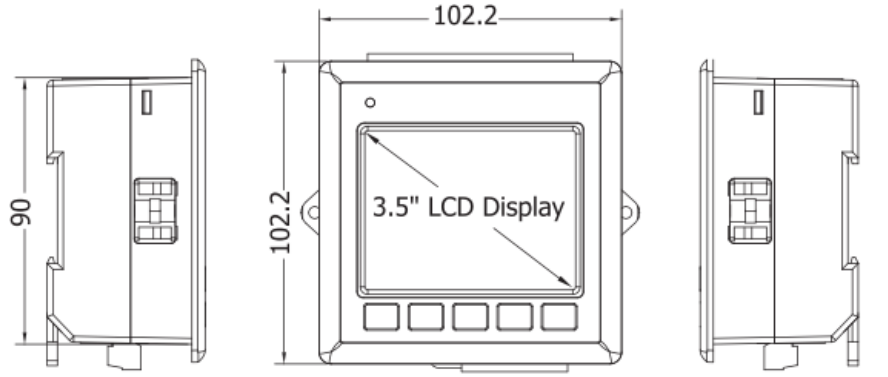
Bottom View



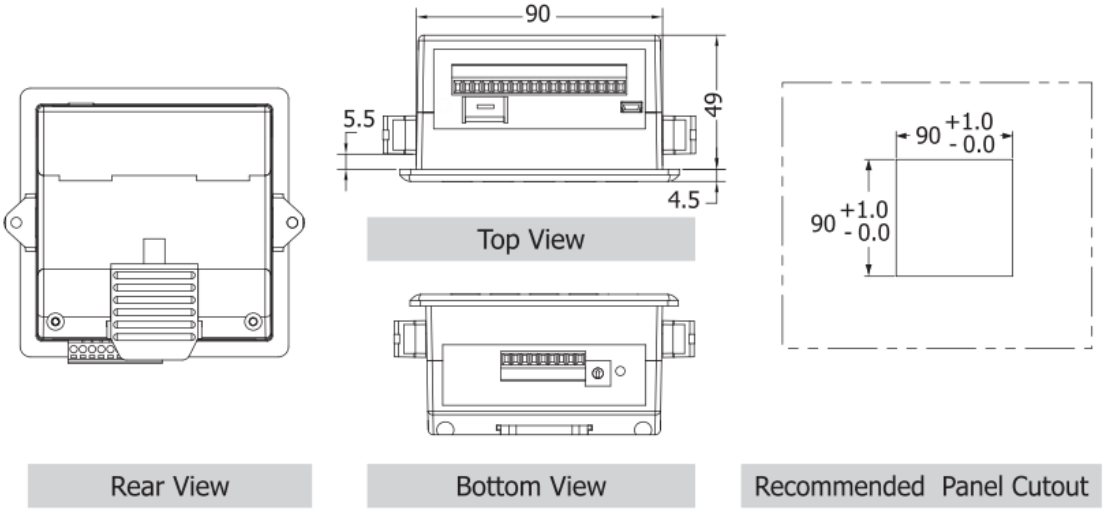
Wall Mounting

VPD-130/VPD-130N (unit: mm)

VPD-130

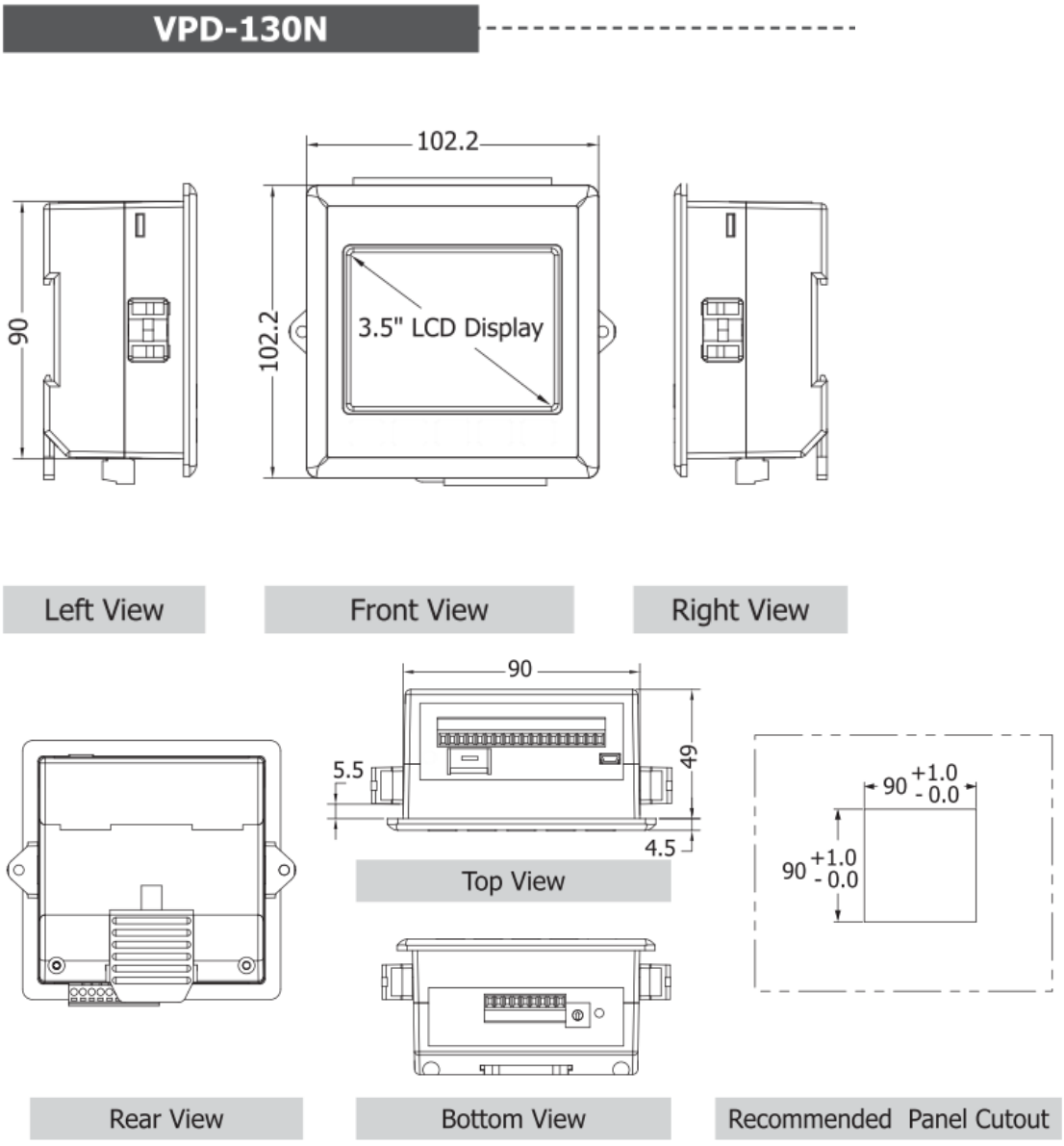


Left View | Front View | Right View



Top View

Rear View | Bottom View | Recommended Panel Cutout



3.3. Installation and Wiring

Mount the hardware

For TPD-280/TPD-280U/TPD-283:

TPD-280/TPD-280U with External Wall Box, EWB-T28 (optional)	TPD-280/TPD-280U with Outlet Box, OB120 (optional)
	

For TPD-430/TPD-430-EU:

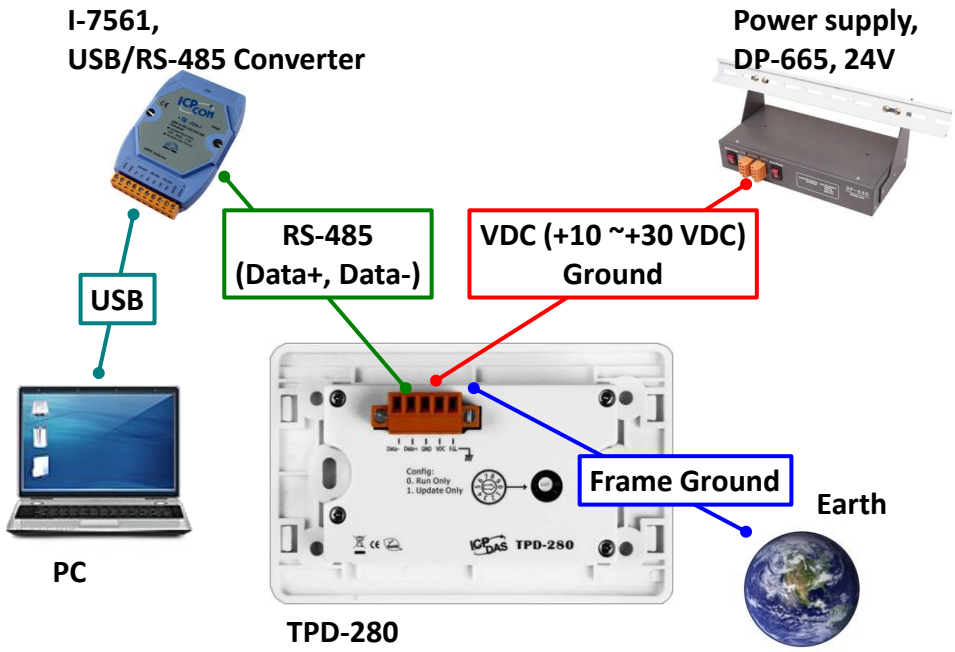
TPD-430 with External Wall Box, EWB-T43 (optional)	TPD-430 with Outlet Box, OB120 (optional)
	
TPD-430-EU for European 86 x 86 mm outlet box	
	

For VPD-130/VPD-130N:

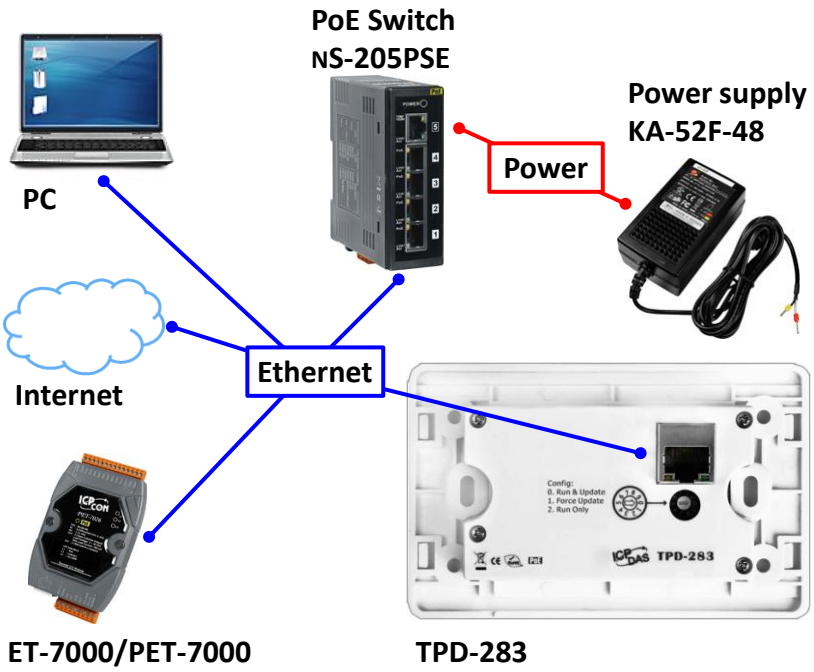
The DIN-rail mounting of VPD-130/VPD-130N	The panel mounting of VPD-130/VPD-130N
	

Connect to power and network

TPD-280



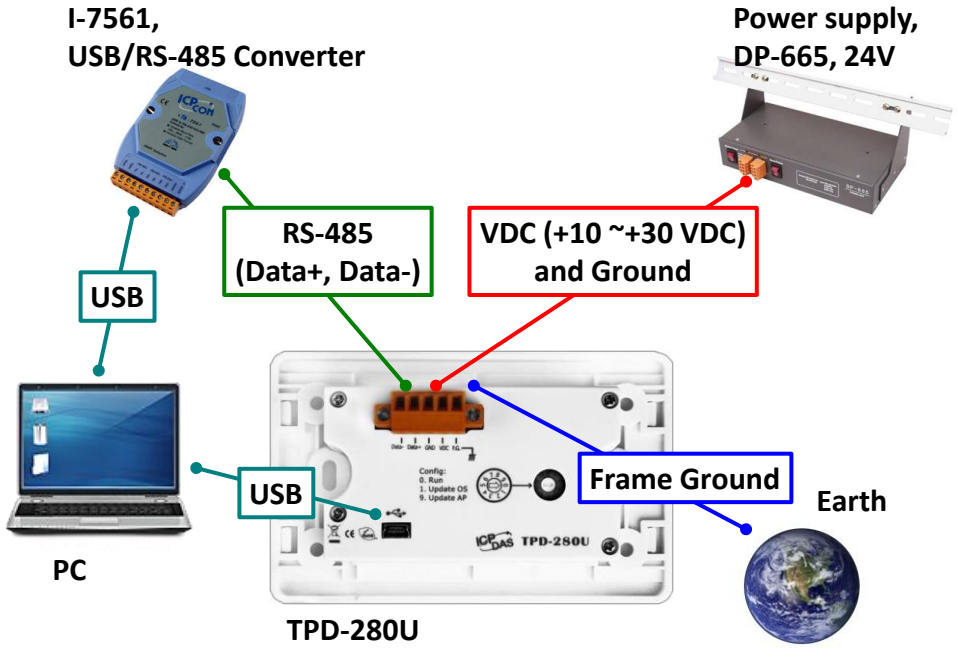
TPD-283



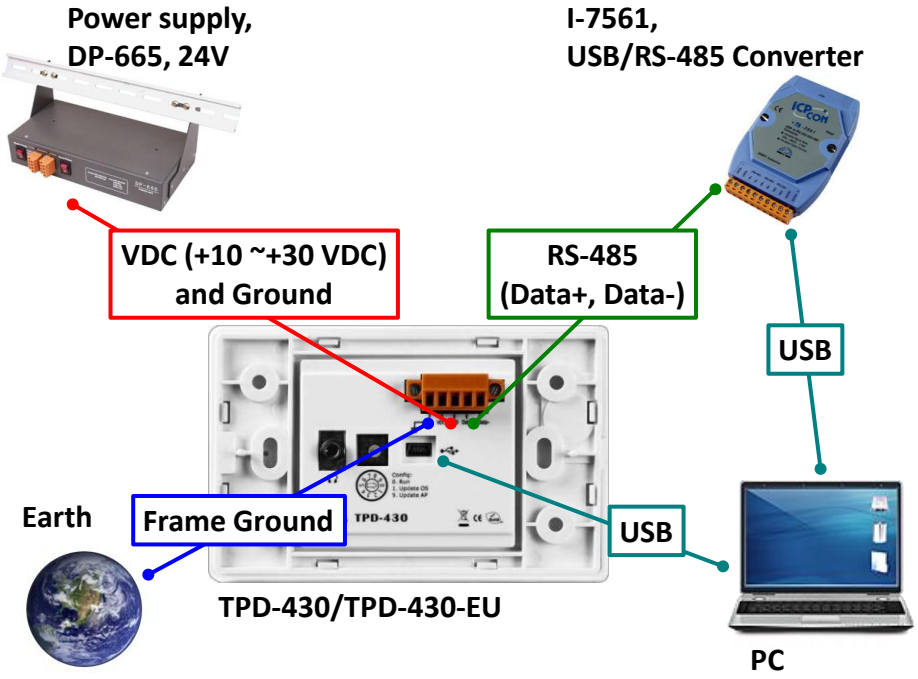
** **Note**: either “Ethernet” or “PoE” uses the same general Ethernet cable.

** PoE, power over Ethernet, means that the Ethernet cable conveys not only data but also power.

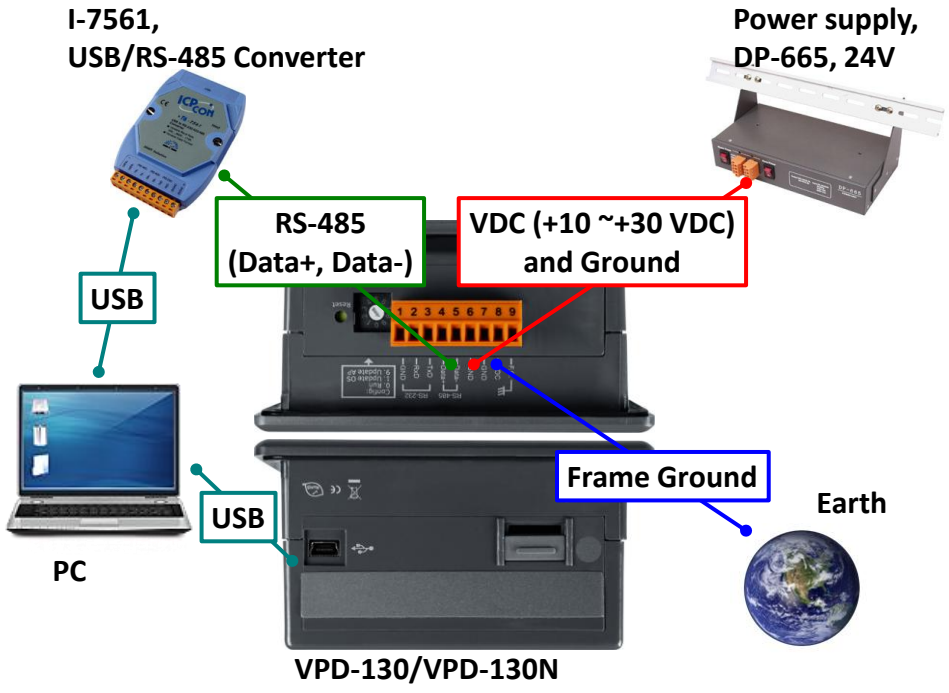
TPD-280U



TPD-430/TPD-430-EU



VPD-130/VPD-130N



The pin assignments

The pin assignments of TPD-280/ TPD-280U/ TPD-283/ TPD-430/ TPD-430-EU:

No	Mode	Description
0	F.G.	Frame Ground. F.G. is connected to the inside EMI or ESD suppression circuits. Make sure that F.G. is connected to the Earth
1	VDC	DC input Voltage (range: +10V ~ +30V)
2	GND	Connected to the power supply's ground pin
3	Data+	The positive data line of the RS-485 network
4	Data-	The negative data line of the RS-485 network

The pin assignments of VPD-130/ VPD-130N:

No	Mode	Description
0	F.G.	Frame Ground. F.G. is connected to the inside EMI or ESD suppression circuits. Make sure that F.G. is connected to the Earth
1	VDC	DC input Voltage (range: +12V ~ +48V)
2	GND	Connected to the power supply's ground pin
3	Data+	The positive data line of the RS-485 network
4	Data-	The negative data line of the RS-485 network
5	TxD	The pin of transmitted data of the RS-232
6	RxD	The pin of received data of the RS-232
7	GND	The common ground of the RS-232

Note: the RS-485 and the RS-232 are the same serial port.

4. Setup Devices and Connect to I/O

This chapter is divided into two parts. One is setup TouchPAD and the other is connecting TouchPAD to I/O modules.

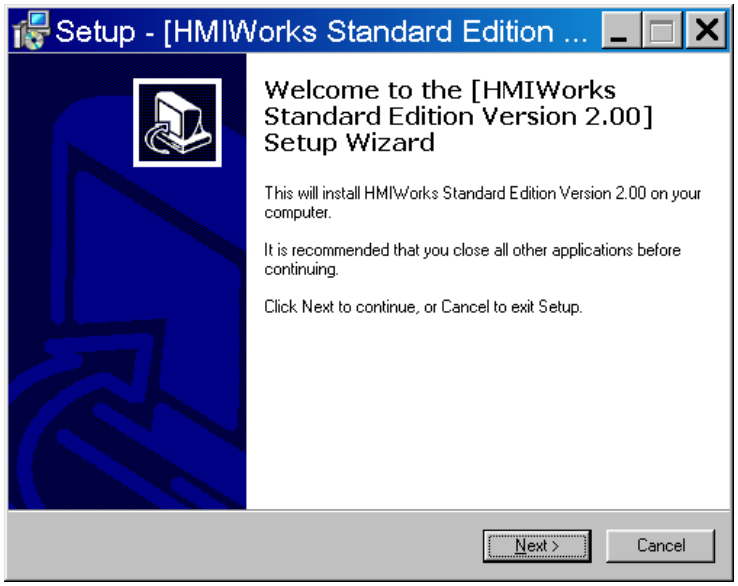
4.1. Preparation

First of all, you should install HMIWorks in your PC. HMIWorks is the development tools for TouchPAD programs.

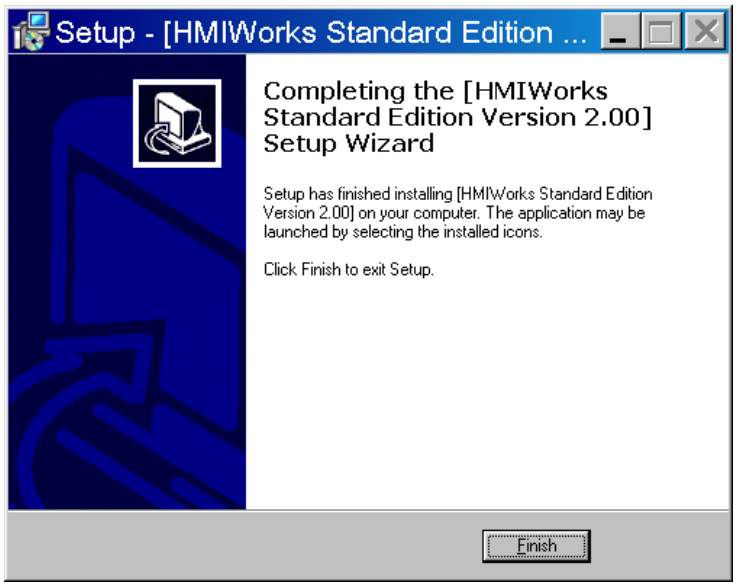
Follow the steps below to install the HMIWorks:

1. Double click the icon to install.

2. Simply follow the instructions to finish the installation.



3. The snapshot of **Finish**.



4.2. Setup Devices

There are several kinds of devices (TouchPADs). Before downloading programs to TouchPAD, TouchPAD must be setup. We describe how to setup each kind of devices in the following sections.

4.2.1. Setup TPD-280

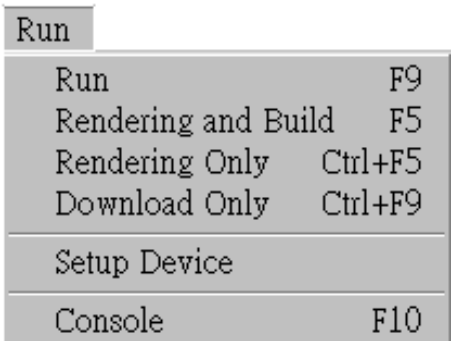
Configuration mode:

In the back of the TPD-280, configuration modes can be found. Use the rotary

switch to set the configuration mode.

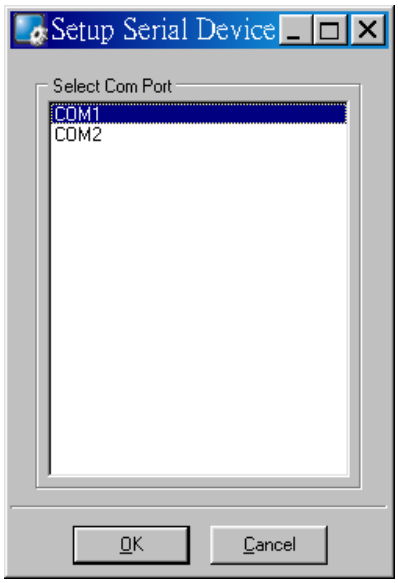
No	Mode	Description
0	Run Only	This mode is used for running programs.
1	Update Only	This mode is used for updating programs.

In the **Run** menu of HMIWorks, Click "**Setup Device**".



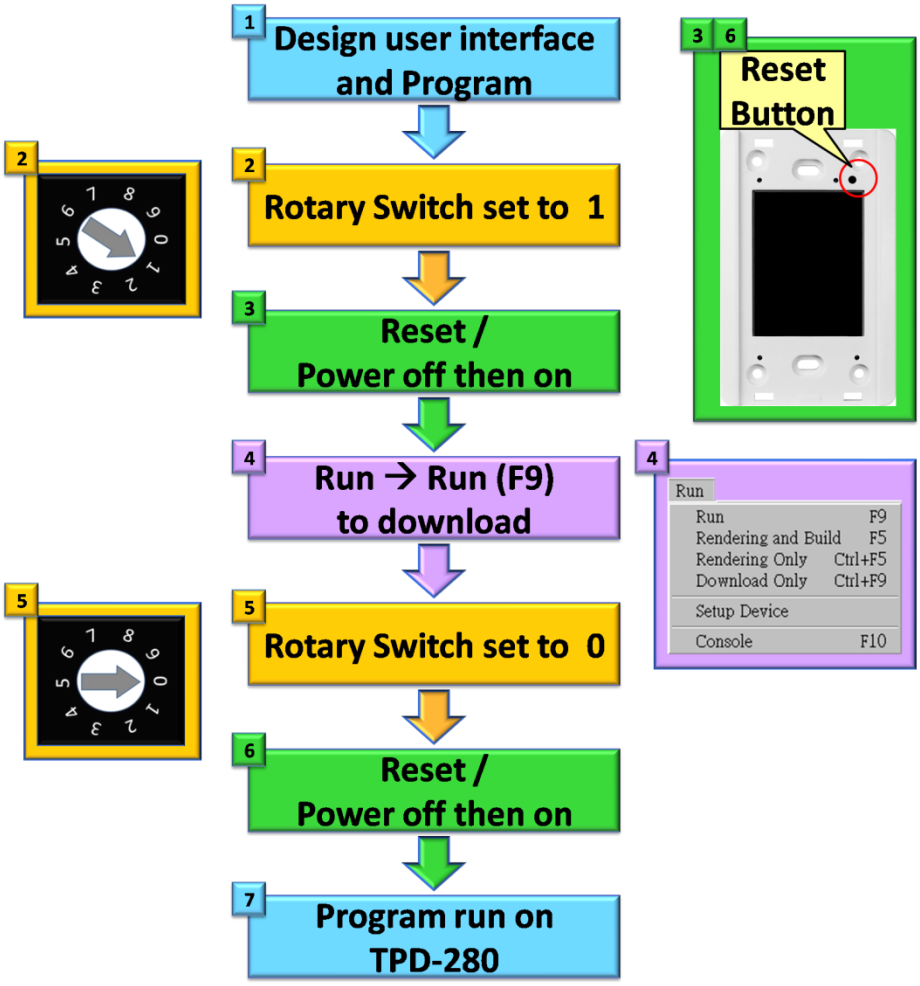
TPD-280 uses RS-485 as its communication method. It is not able to connect through Ethernet.

Select the com port that connects to the TPD-280.
Before downloading programs to the TouchPAD, be sure to setup the TouchPAD (TPD-280) to connect first.



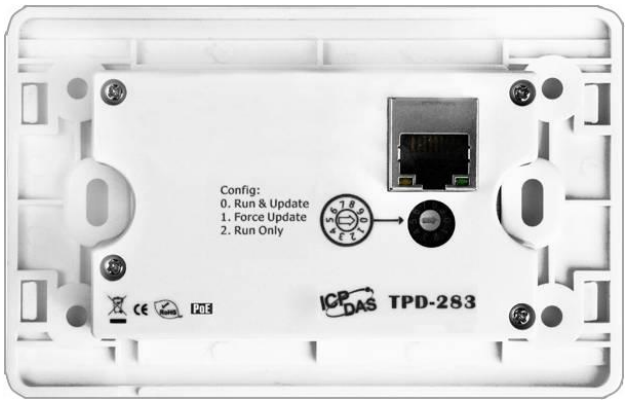
Download programs to TPD-280

Downloading programs to TPD-283 is easy. Just click on “**Run**” or “**Download Only**”. However, downloading programs to TPD-280 is a little complicated. As the following figure shows, set rotary switch to “1” when downloading the program and set rotary switch to “0” to let the program run.



4.2.2. Setup TPD-283

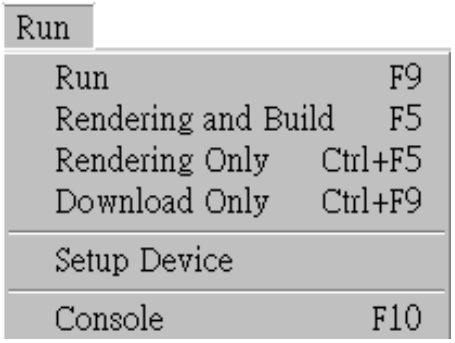
Configuration mode:



In the back of the TPD-283, configuration modes can be found. Use the rotary switch to set the configuration mode.

No	Mode	Description
0	Run & Update	This is a special run mode which is used in the development phase. TouchPAD can be updated by remote side PC through Ethernet.
1	Force Update	While program run on the TouchPAD seriously crashes, use this mode to update program to TouchPAD.
2	Run Only	Simply run, TouchPAD cannot be updated in this mode.

In the **Run** menu of HMIWorks, Click "**Setup Device**".



TPD-283 uses Ethernet as its communication method. It is not able to connect through RS-485 network (since it doesn't have).

Before downloading programs to the TouchPAD (TPD-283), be sure to setup the TouchPAD to connect. **TPD-283's IP configuration is not stored directly in the memory of itself but is a part of the program which is**

compiled by HMIWorks. Every time HMIWorks downloads a program to TPD-283, it also updates the IP settings of TPD-283.

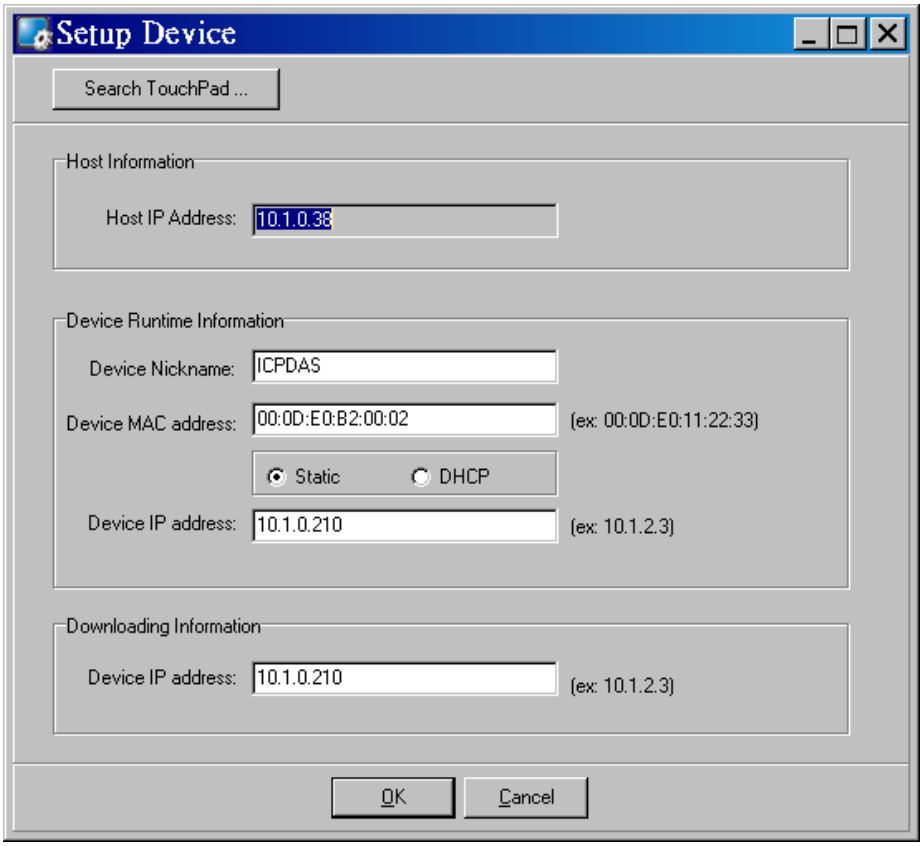
Default IP settings of TPD-283

Item	Value
IP	192.168.255.1
Type	Static

The IP settings of a TPD-283 are in one of the two types: DHCP or Static.

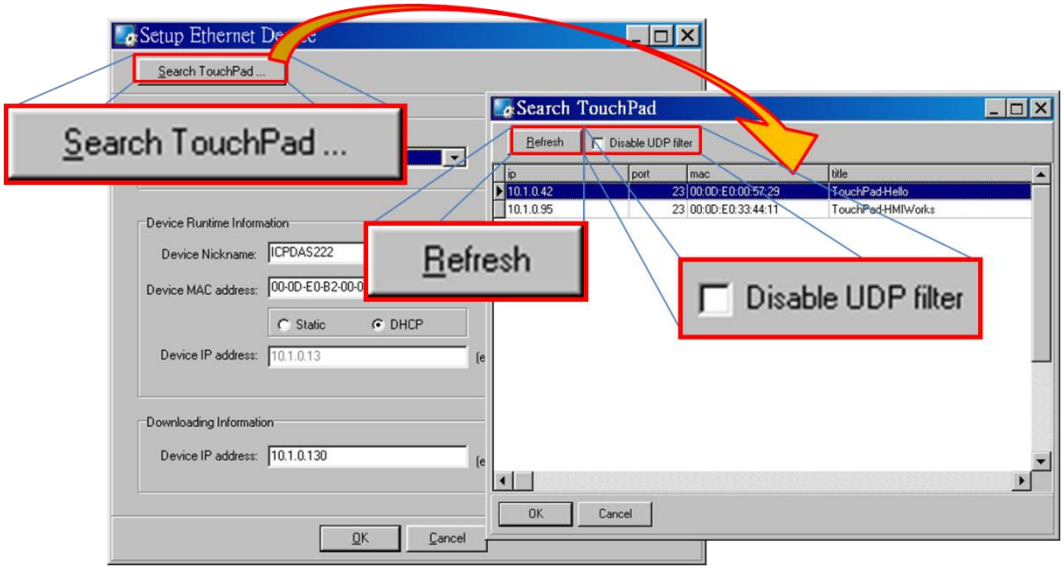
Type	Description
DHCP	TPD-283 is assigned an IP from the DHCP server every time it powers on. Note that presently TPD-283 **blocks as a blank screen** when turning on the power of the TPD-283 with DHCP enabled. This is because of continuously waiting for DHCP's IP assignment.
Static	TPD-283 has a static, unchanged IP.

Users can change the type of IP settings by downloading a new program into TPD-283. **Run → Setup Device,** and click the button, **"Search TouchPad …",** first. Clicking on the searched TPD-283 shows the "Setup Ethernet Device" window. Choose the IP type, DHCP or Static, for the new program and then press **OK**. **Run → Run** to download the new program. (We introduce the details in the below.) Note that the new IP settings take effect after successful downloading because the IP settings are parts of the program.



item	description
Search TouchPAD	Search TouchPADs on the network. Make sure that TouchPADs and the local computer are in the same subnet.
Host IP Address	The IP address of the local computer
Device Nickname	The nickname used to identify the TouchPAD which is selected in the list of the "**Search TouchPAD**" window. This nickname is part of the program and it takes effect after the new program runs (downloading successful).
Device MAC address	The MAC address of the TouchPAD which is selected in the "**Search TouchPAD**" window. Every TouchPAD is shipped with its MAC information pasted on its back.
Device IP Address (in the runtime)	The IP address of the TouchPAD which is used in the runtime of the program. The IP address in the runtime is part of the program and it takes

	effect after the new program runs (downloading successful).
Device IP Address (for downloading)	The IP address of the TouchPAD which is used only when downloading programs. This IP information is not part of the program, every time the downloading process starts, HMIWorks assigns this IP just for downloading.



item	description
Refresh	Re-search again to re-make the list.
Disable UDP filter	We have UDP filter enabled by default to search TouchPAD only. Devices which are not TouchPAD are filtered out if this option is not checked.

Press **"Search TouchPAD"** button to search TPD-283.

Note1: MAC address can be found in the back of TPD-283.
HMIWorks uses MAC to search TouchPAD. To see if TouchPAD is on the network, check that if there is a device with the MAC in the list after searching.

Note2: Be sure to put TPD-283 and your PC in the same subnet.

What to do if PC and TouchPAD are not in the same subnet?

➔ **For the cases of not knowing TouchPAD's IP.**

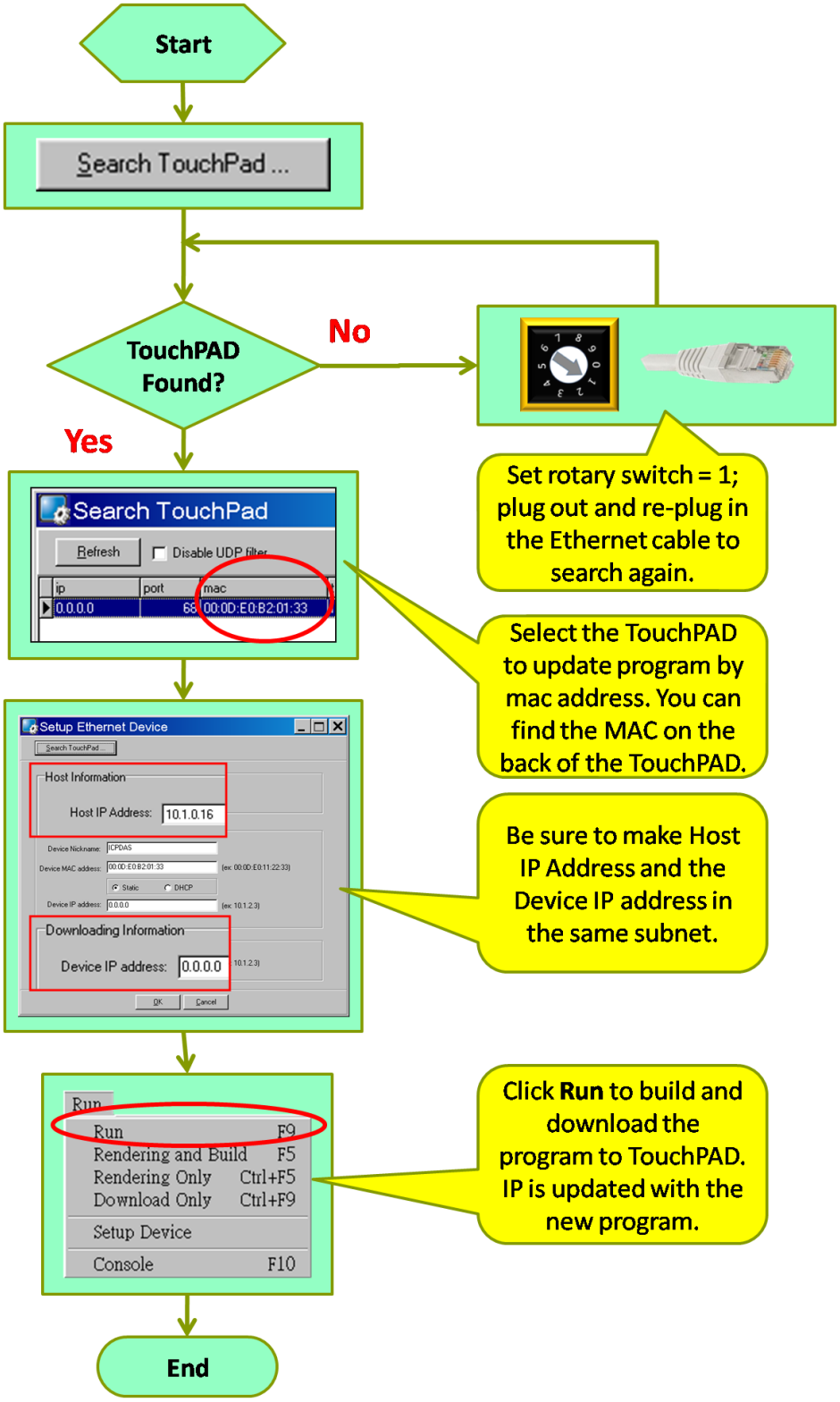
(Normally we set rotary switch of TouchPAD to zero.)

<table>
<tr><th></th><th>The IP settings of type DHCP is currently on TouchPAD</th><th>The IP settings of type Static is currently on TouchPAD</th></tr>
<tr><td rowspan="2">TouchPAD and PC are in the same subnet</td><td>DHCP server is connected with TouchPAD:
“Search TouchPad …” and downloading are both successful.</td><td rowspan="2">“Search TouchPad …” and downloading are both successful.</td></tr>
<tr><td>DHCP server is not connected with TouchPAD:
Specially set rotary switch of the TouchPAD to 1, “Search TouchPad …”, click on the searched TouchPAD and finally specify an IP (in the same subnet as PC) in the field of “Device IP address” in the “Setup Ethernet Device” window. Run → Run to download the new program with the rotary switch still set to 1.</td></tr>
<tr><td rowspan="2">TouchPAD and PC are in the different subnet</td><td>DHCP server is connected with TouchPAD:
The same step as the case of disconnected DHCP server and both PC and TouchPAD are in the same subnet.</td><td rowspan="2">“Search TouchPad …” is successful, click on the searched TouchPAD and finally specify an IP (in the same subnet as PC) in the field of “Device IP address” in the “Setup Ethernet Device” window. Run → Run to download the new program.</td></tr>
<tr><td>DHCP server is not connected with TouchPAD:</td></tr>
</table>

	The same step as the case of disconnected DHCP server and both PC and TouchPAD are in the same subnet.	

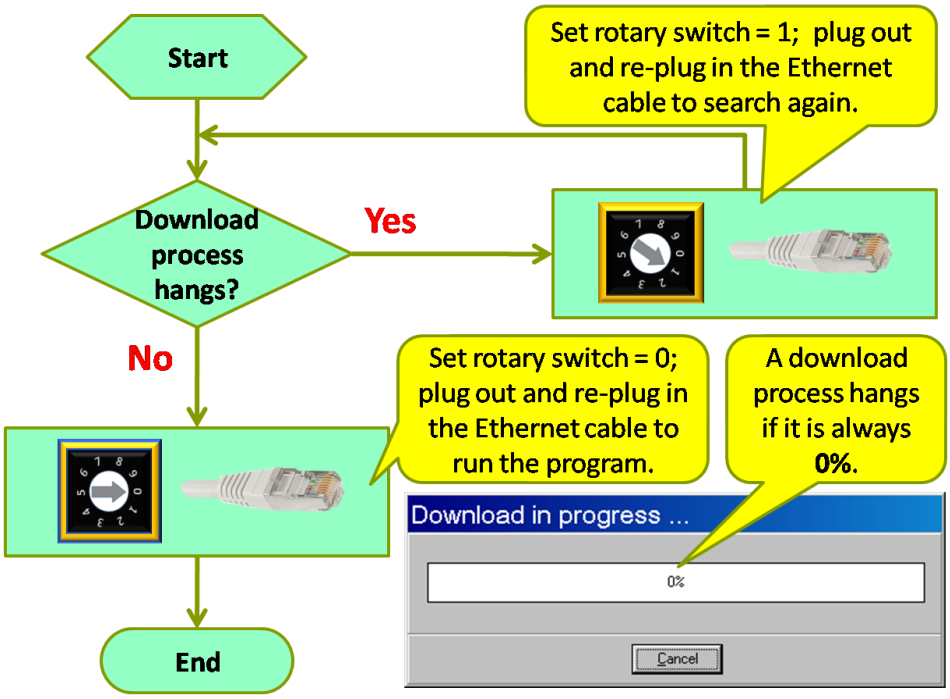
Note that the new IP settings take effect after successful downloading because the IP settings are parts of the program.

In short, we draw the flow chart of setup TPD-283:



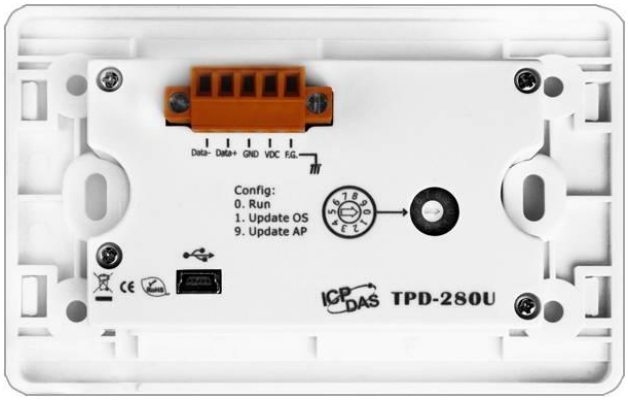
What to do if download process hangs?

Anytime download process hangs, users can follow the flow below to finish the download process. (Note that below is not just for the case after setup device.)



4.2.3. Setup Other Devices in TouchPAD Series

Configuration mode:



Except TPD-280 and TPD-283, users can find out the same information of configuration modes on their shells of other devices in the TouchPAD series. We use the rotary switch to set the configuration mode. Take TPD-280U for

example as below.

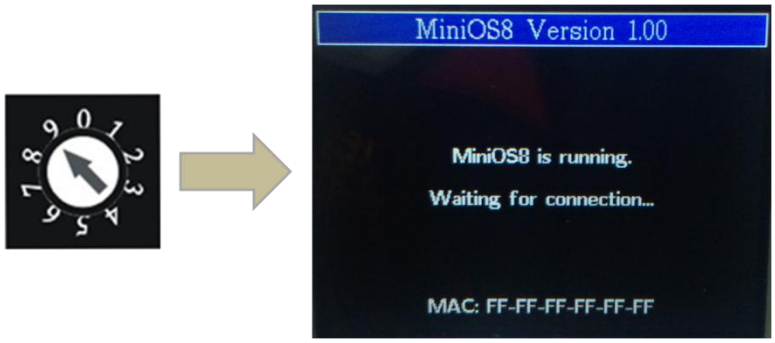
No	Mode	Description
0	Run	This mode is used to run the application. (There is only one application on a TouchPAD.)
1	Update OS	This mode is used to update operating system of TouchPAD.
9	Update AP	This mode is used to download applications to TouchPAD. (There is only one application on a TouchPAD.)

All devices in the TouchPAD series have USB ports, except TPD-280 and TPD-283. HMIWorks can download programs through that USB port. Unlike TPD-280 and TPD-283, Users **need not** “Setup Device” in the **Run** menu but users must install USB driver in their PC first instead.
Note: Other means of downloading are not provided for TouchPAD with USB on itself.

Install TouchPAD USB Driver

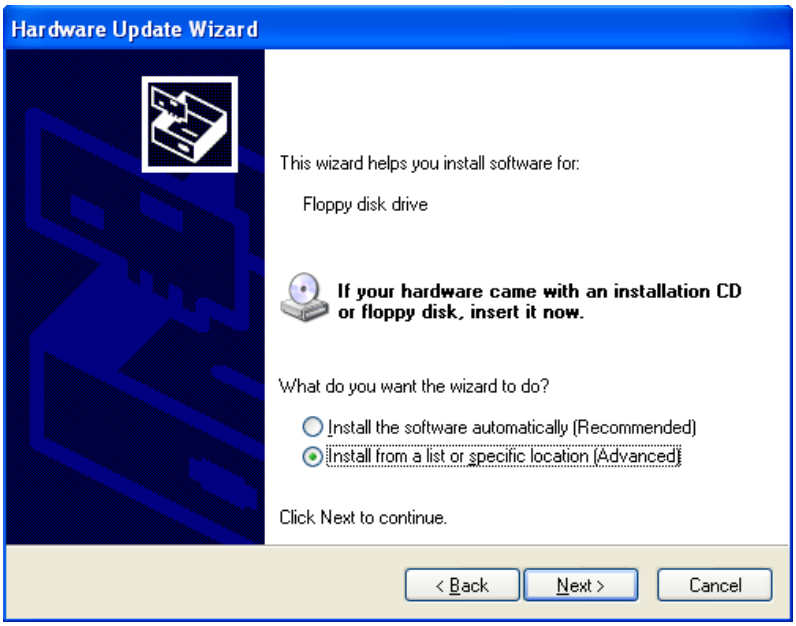
This section describes how to install HMIWorks TouchPAD DFU (Device Firmware Update) Driver. To install DFU driver allows user to download programs to TouchPAD via USB port. Follow these steps to install the driver: (These steps are written for TouchPAD which have USB ports, such as TPD-280U, TPD-283U, TPD-430, TPD-433, etc.)

1. Using USB cable to connect to the TouchPAD Device. Rotary Switch set to “9” and you can see the screen of update mode.

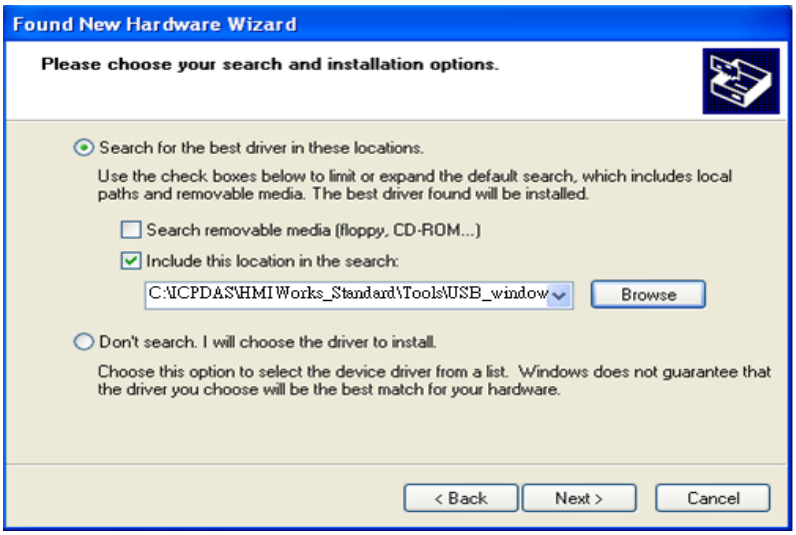


2. The ***Found New Hardware Wizard*** pops up thereafter, and select “No, not this time” and press “**Next**”.

3. Select “Install from a list or specific location (Advanced)” and press “**Next**”.



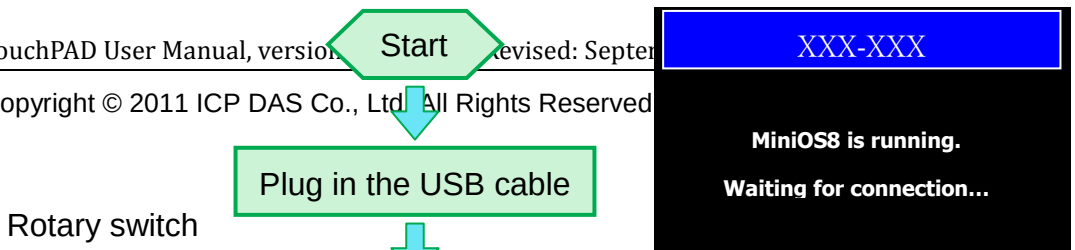
4. Select “Include this location in the search”. Browse to the location of the driver files and press “**Next**”.
   The default path for the driver is:
   C:\ICPDAS\HMIWorks_Standard\Tools\USB_windows_drivers
   , where “C:\ICPDAS\HMIWorks_Standard\” is the installation path.



5. Finally, the Device Firmware Upgrade driver is installed successfully.

Download programs through USB

Before downloading programs to the TouchPAD, be sure to install the HMIWorks TouchPAD DFU (Device Firmware Update) Driver first. Then remember to plug the USB cable into the TouchPAD.
Follow the flow to download a program to TouchPAD.



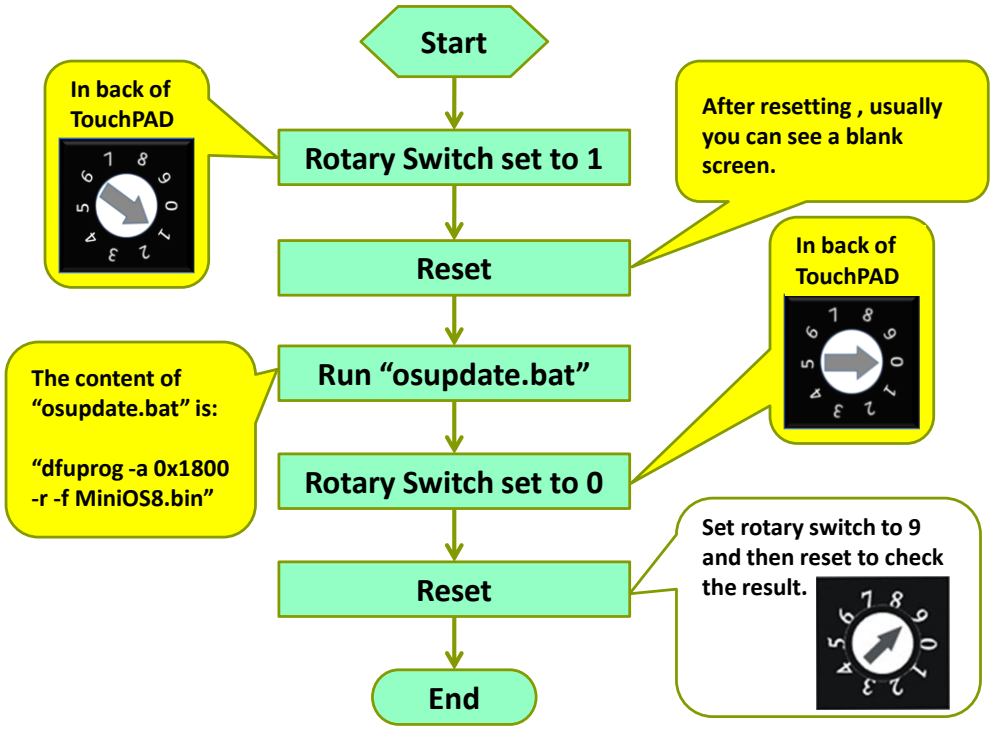
Update OS through USB

Except TPD-280 and TPD-283, all other devices in the TouchPAD series have an OS on them. In cases that users may need to update OS, we introduce the flow to do that.

Before updating OS (Operating System) to the TouchPAD, be sure to install the HMIWorks TouchPAD DFU (Device Firmware Update) Driver first. Then remember to plug the USB cable into the TouchPAD.

You can find that an OS image file is in the following directory: [HMIWorks_Install_Path]\bin\boot\[Device_Name]\MiniOS8.bin. For example, an OS image file, MiniOS8.bin can be found at "C:\ICPDAS\HMIWorks_Standard\bin\boot\TPD-280U\" if trying to find out TPD-280U's OS image file. And in the same directory, there's a batch file which is used to update OS and it is called "osupdate.bat".

Follow the flow to update OS to TouchPAD.



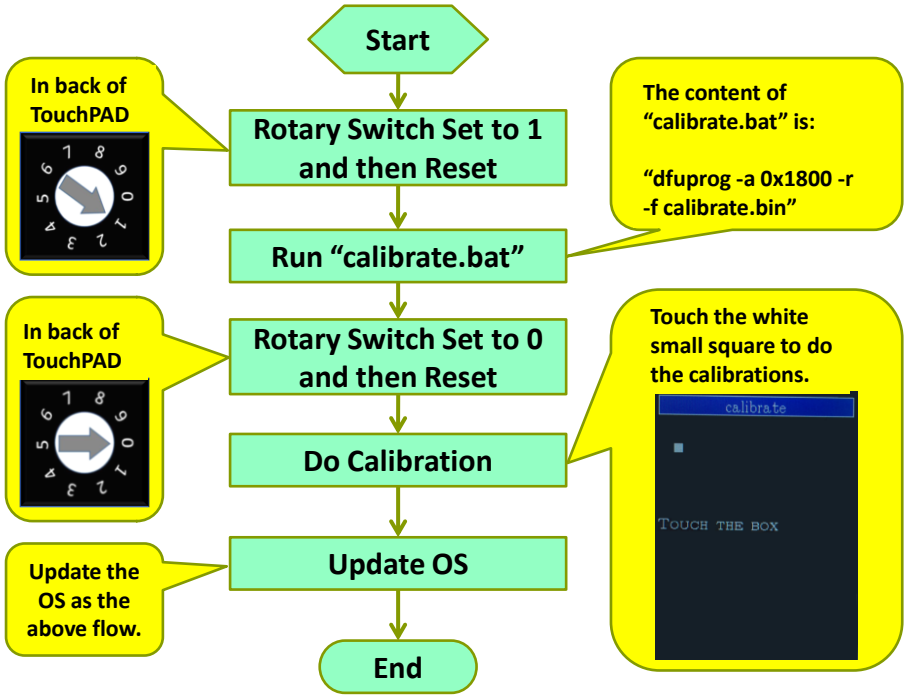
Do Calibrations

Usually users need not to calibrate the touch screens, we calibrate the TouchPADs before shipping. However, in cases users may need to do calibrations, we introduce the flow here.

Before downloading the calibration program to the TouchPAD, be sure to install the HMIWorks TouchPAD DFU (Device Firmware Update) Driver first. Then remember to plug the USB cable into the TouchPAD.

You can find that the calibration programs are in the following directory: "[HMIWorks_Install_Path]\bin\boot\[Device_Name]\calibrate". For example, there are two calibration programs can be found at "C:\ICPDAS\HMIWorks_Standard\bin\boot\TPD-280U\calibrate" if trying to calibrate TPD-280U. One is for landscape (when the screen is horizontal) and the other is for portrait (when the screen is upright). And in the same directory of the file, calibrate.bin, there's a batch file which is used to download the calibrate.bin to the TPD-280U and it is called "calibrate.bat".

Follow the flow to calibrate TouchPAD.

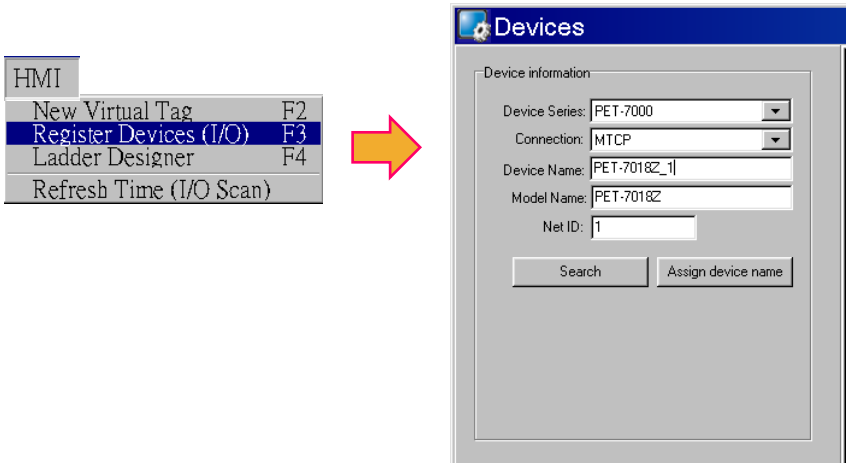


4.3. Connect to I/O Modules

We provide connection methods for three series of I/O modules, PET-7000, I-7000, and M-7000 and a general method for Modbus TCP I/O modules. All these I/O modules have the same model for connections in general.
They need to

1. create a new connection first (in the **Workspace**)
2. import the tags for the specified I/O module (while registering device)

Below we introduce the "**Devices**" window which is called while registering device (**F3**).



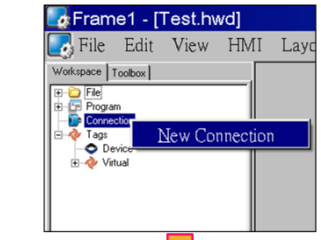
A *valid Device or Connection name* is a sequence of one or more letters, digits or underscore characters (_). It must not begin with a digit.

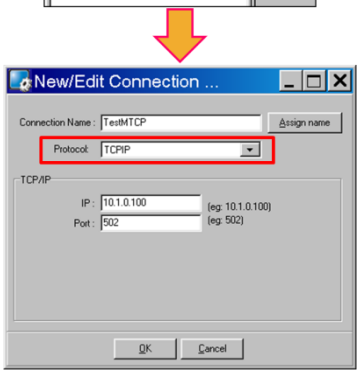
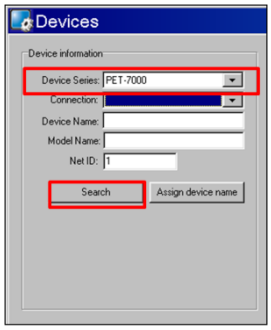
Item	Description
Device Series	Specify the series which the I/O module to connect belongs to. (For example, PET-7000, I-7000, Modbus TCP, etc.)
Connection	Specify the connection methods, TCPIP or UART, to connect to the I/O module.
Device Name	Specify the name of the I/O module. Users can assign a name they want.
Modal Name	Specify the modal of the I/O module to connect.
Net ID	The specified ID of the I/O module in the network. *Possible range: 0 ~ 255.

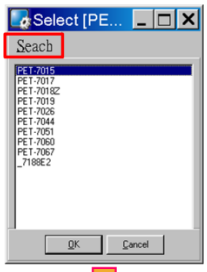
*note: for PET-7000 series, the possible range of Net ID is 1 ~ 255.

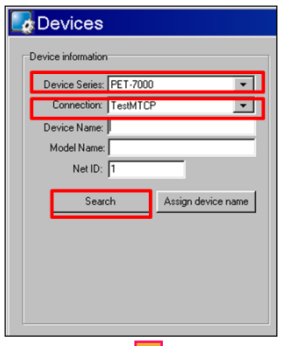
Create the Connections and Import Tags

	PET-7000 series I/O by Search	PET-7000 series I/O by Select
Create the connections	(No Need)	Create a new connection through protocol “TCPIP”. Here we set “Connection Name” as “TestMTCP”.

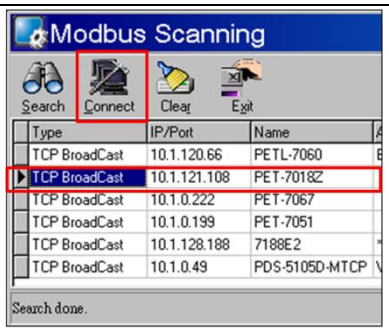
<table>
<tr>
<td></td>
<td></td>
<td> 


</td>
</tr>
<tr>
<td>Import tags</td>
<td>Press F3 to register device. Choose “Device Series” as “PET-7000” and press “Search”.
 

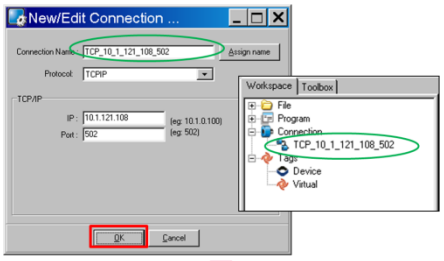
Press “Search”.
 

Click the I/O module you want and press “Connect”.</td>
<td>Press F3 to register device. Choose “Device Series” as “PET-7000”, choose “Connection” as “TestMTCP” which you created in the previous step and press “Search”.
 

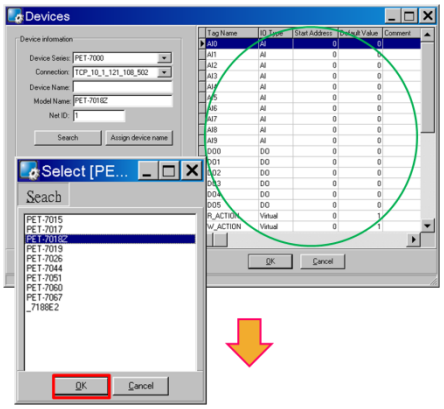
Click the I/O module you want and then press “OK” to add tags of the specified I/O module.</td>
</tr>
</table>



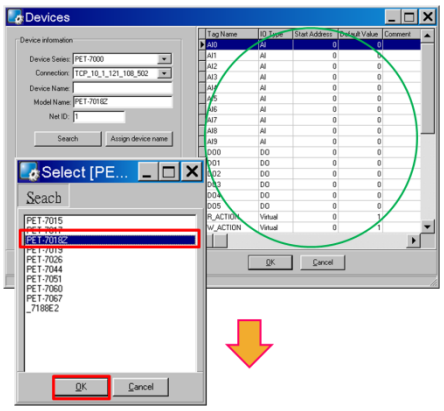
Press “**OK**” and then you can see a new connection is added in the **Workspace**.



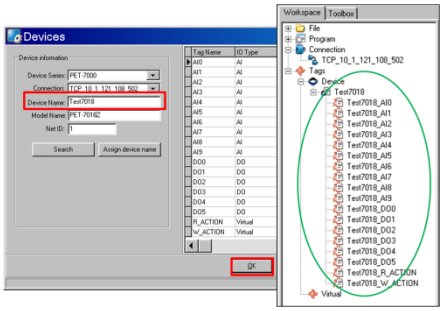
Press “**OK**” to add tags of the specified I/O module.

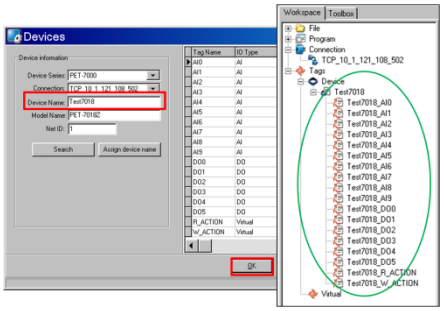


Assign the “Device Name”, press “**OK**” and then you can see the I/O module is added to the **Workspace**.

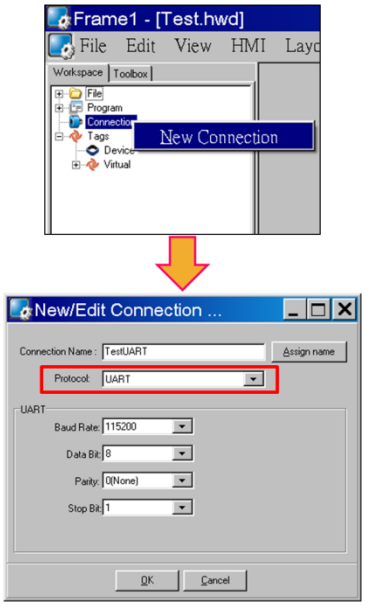
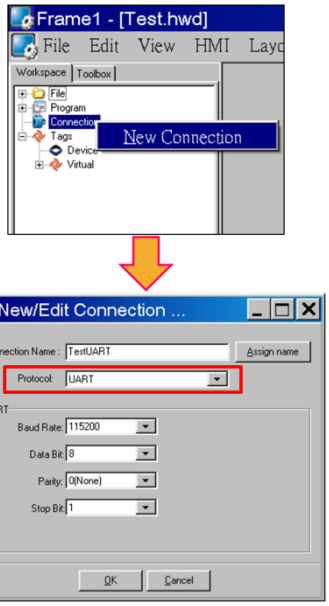


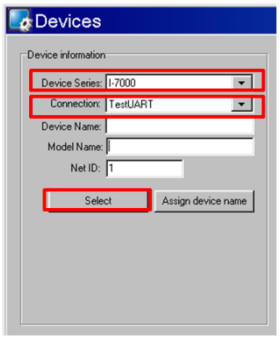
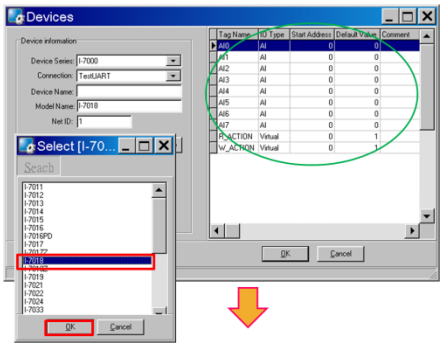
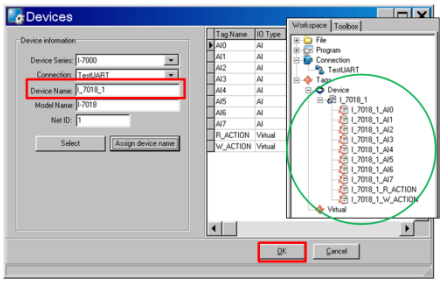
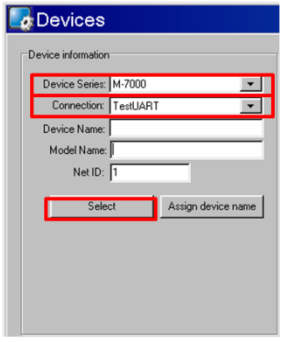
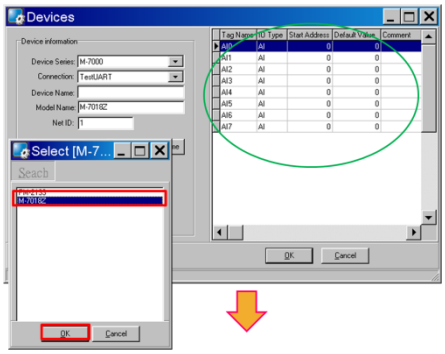
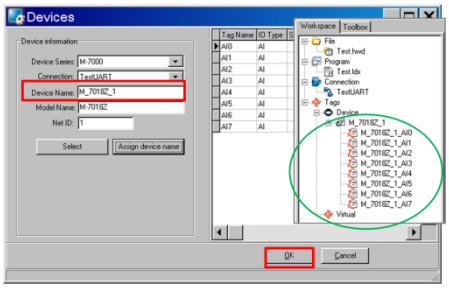
Assign the “Device Name”, press “**OK**” and then you can see the I/O module is added to the **Workspace**.



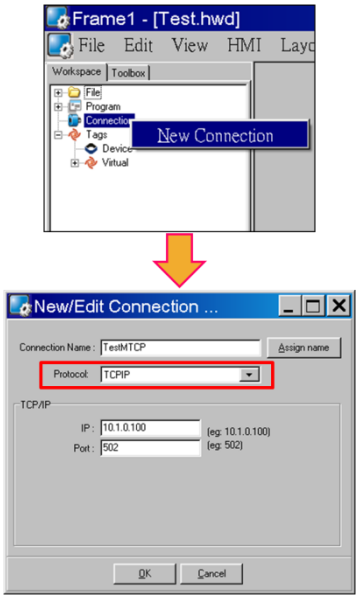
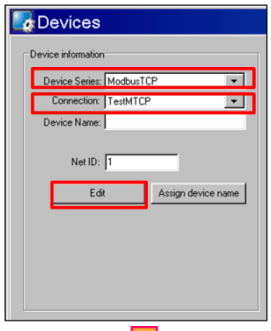
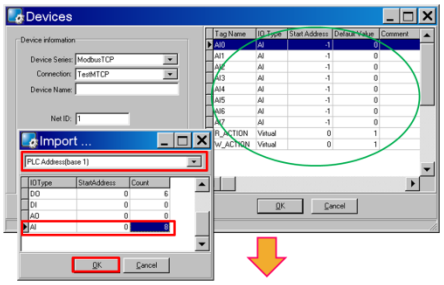
Create the Connections and Import Tags (continued)

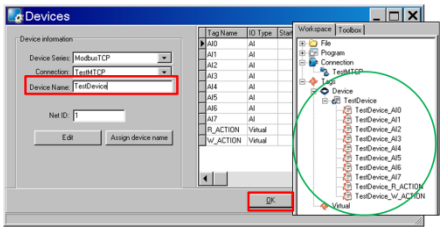
	I-7000 series I/O by Select	M-7000 series I/O by Select
Create the connections	Create a new connection through protocol “UART”. 	Create a new connection through protocol “UART”. 
Import tags	Press **F3** to register device. Choose Device Series as “I-7000”, choose “Connection” as “TestUART” which you created in the previous step and press “**Select**”.	Press **F3** to register device. Choose Device Series as “M-7000”, choose “Connection” as “TestUART” which you created in the previous step and press “**Select**”.

	 Click the I/O module you want and then press "**OK**" to add tags of the specified I/O module.  Assign the "Device Name", press "**OK**" and then you can see the module is added to the **Workspace**.  	 Click the I/O module you want and then press "**OK**" to add tags of the specified I/O module.  Assign the "Device Name", press "**OK**" and then you can see the module is added to the **Workspace**. 

Create the Connections and Import Tags (continued)

	Modbus TCP I/O by Edit	
Create the connections	Create a new connection through protocol "TCPIP".	

		
Import tags	Press **F3** to register device. Choose Device Series as “ModbusTCP”, choose “Connection” as “TestMTCP” which you created in the previous step and press “**Edit**”.  In the **Import** window, choose the addressing mode, set the numbers of I/O points and then press “**OK**” to add the tags of the Modbus TCP I/O module. 	

	Assign the “Device Name”, press “**OK**” and then you can see the module is added to the **Workspace**. 	

5. Development Software, HMIWorks

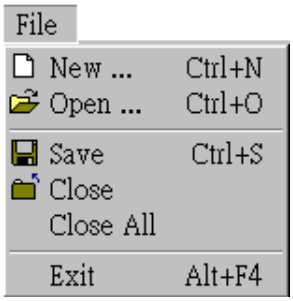
HMIWorks is the development tool for, both TPD and VPD series. It supports two programming types, Standard C and Ladder. Compared with traditional GUI development tools, HMIWorks is easy to learn, flexible to design GUIs, and takes less time to raise productivity.

Features of HMIWorks include:

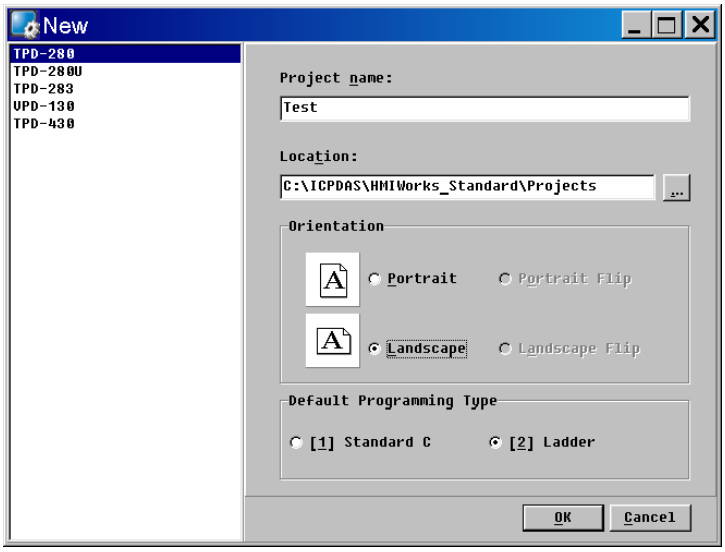
- FREE of charge (for ICP DAS TouchPAD devices)
- Two programming types, ladder diagram and Standard C
- Plenty of widgets
- Plenty of demos shorten development time
- Advanced search for I/O modules
- Detail error messages
- Easy downloading after building
- Automatic generated codes for user-designed frames
- Multi-frame design
- WYGIWYS (What You Get Is What You See)
- Abstract graphics as simple APIs
- Easy learning IDE to raise productivity in short time

5.1. The Construction of HMIWorks

Before showing the construction of HMIWorks, create a new project first.



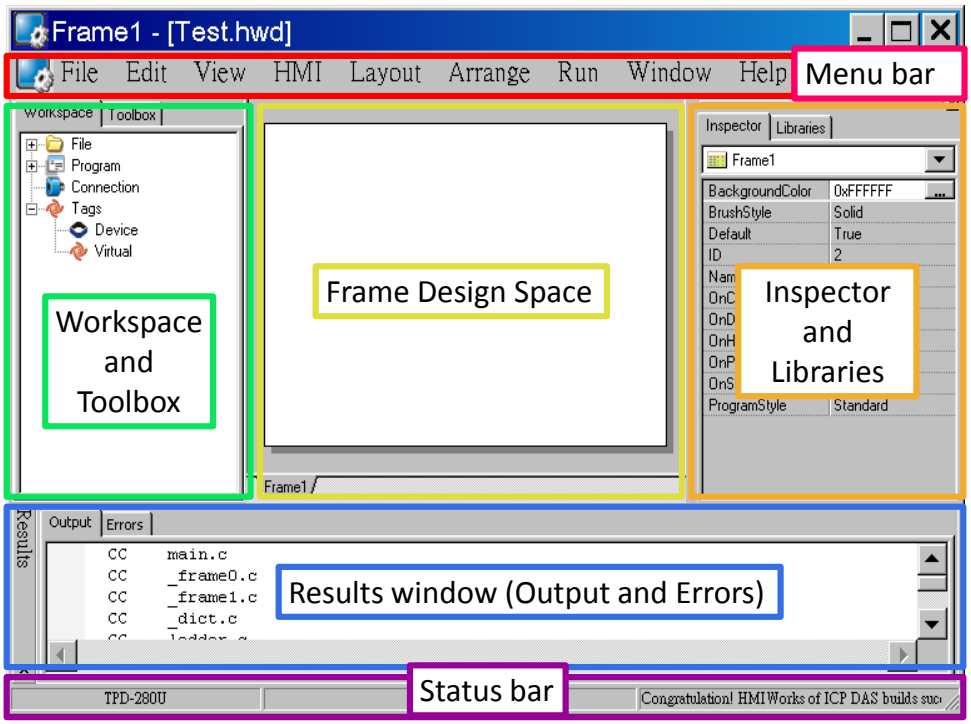
Click **File** menu, then click on **New…**.



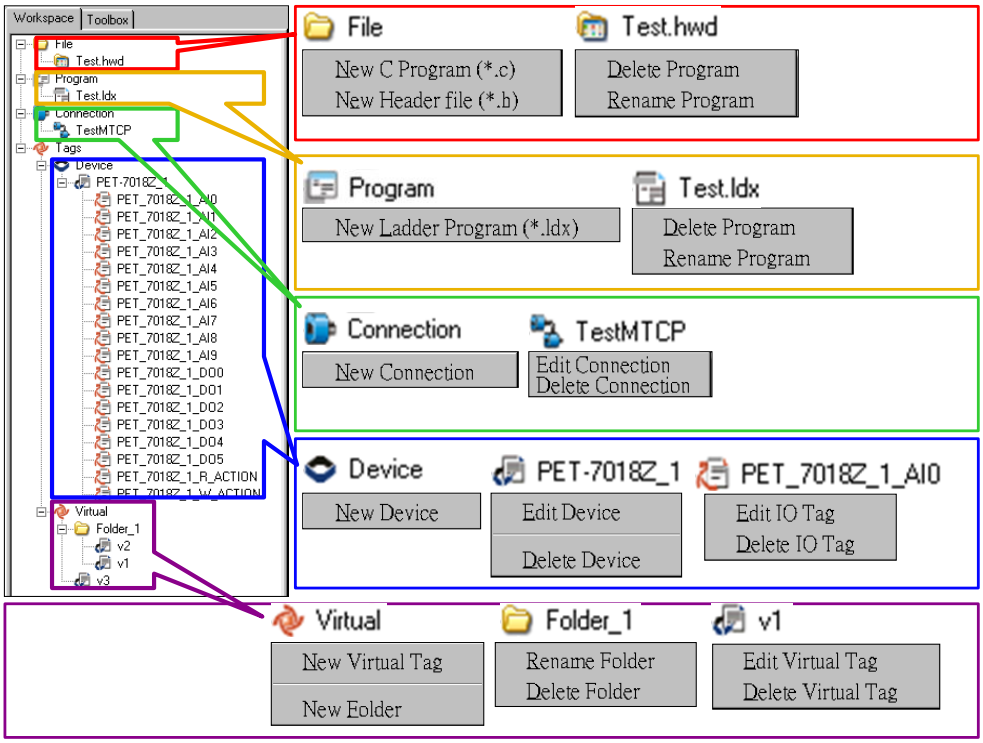
A *valid project name* is a sequence of one or more letters, digits or underscore characters (_). It must not begin with a digit. Besides, it is of suggested length 100 characters (including its path).

Choose the target module, **Orientation**, and the **Default Programming Type**. Press **OK** and HMIWorks integrated design environment shows as below. There are several parts of HMIWorks.

1. Menu bar
2. Workspace and Toolbox
3. Frame Design space
4. Inspector and Libraries
5. Results window (Output and Errors)
6. Status bar



Below are the operations (pop-up menus) that users have in Workspace.



Next sections show the functions of these parts.

5.2. The Options of TouchPAD

Users can set the options of the TouchPAD in the file, hmi_options.h.
The file is located in “C:\ICPDAS\HMIWorks_Standard\include” directory, where “C:\ICPDAS\HMIWorks_Standard\” is the installation directory.

The defines in the hmi_options.h are the options for TouchPAD. They are listed below:

Define	Default	Descriptions
HMI_STARTUP_DELAY	0	This is used to delay the system for a while before enabling the LCD backlight. (unit=ms)
HMI_WDT_ENABLE	0	Normally, the system will automatically refresh the watchdog timer. And the system should be rebooted only when a procedure is blocking the system for a period larger than the HMI_WDT_TIMEOUT value.
HMI_WDT_TIMEOUT	2000	The timeout value to reboot (Valid range: 1,000 ~ 50,000 ms)
HMI_WDT_CLEAR	500	The refreshing period to prevent reboot. The system refreshes (clears) the watchdog timer to prevent rebooting.
HMI_STARTUP_BEEP	1	When startup, 1 = Beep, 0 = disable
HMI_TOUCH_BEEP	1	When touched, 1 = Beep, 0 = disable
HMI_BEEP_FREQ	800	For TPD-430/ TPD-430-EU only. This is the pitch value of the beep. The valid range is 30 ~ 4,000 Hz.
HMI_BEEP_MS	25	For TPD-430/ TPD-430-EU only. The time length for each beep
HMI_LCD_BRIGHTNESS	255	For TPD-430/ TPD-430-EU only. 0=the darkest, ..., 255=the brightest.
HMI_STARTUP_LED	0	1=Enable red LED indicator, 0 = Disable red LED indicator. This option is used for TPD-430 and VPD-130 only.
HMI_LCD_AUTO_OFF	30	How long does it take to turn the LCD off

		automatically when the touch is idle. The LCD will back again when touched is pressed and released. 0 = disable, 5 ~ 300 seconds
HMI_LCD_ON_BEEP	1	0 = disable, 1 = Beep when LCD backlight is turned on
HMI_LCD_OFF_BEEP	1	0 = disable, 1 = Beep when LCD backlight is turned off

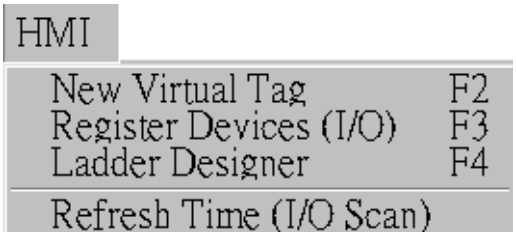
Note:
The options in the hmi_options.h are treated as global settings that affect all projects. To affect only one project, copy this hmi_options.h file into the project directory to modify.

5.3. Ladder Designer

One of the most important features of HMIWorks is Ladder Designer.
The ladder logic is defined by the followings:

1. A Ladder Diagram consists of many rungs.
2. Each rung resembles a circuit which is formed by relays.
3. All of the rungs are executed serially in a loop.

Click **HMI** menu to use this feature.



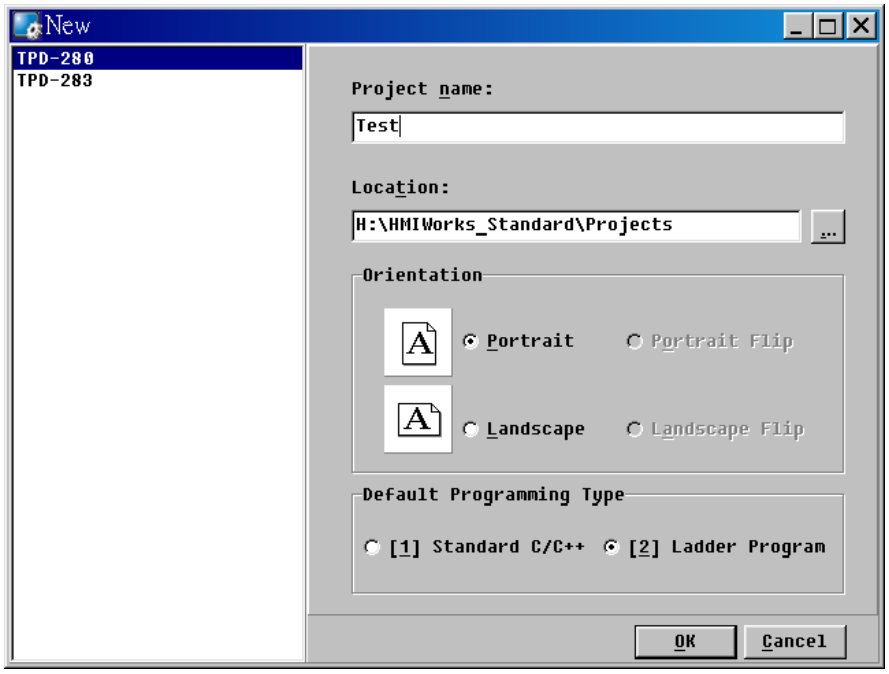
- **New Virtual Tag**: defines your own variables
- **Register Devices (I/O)**: uses I/O devices of ICP DAS on the networks
- **Ladder Designer**: designs your ladder logics
- **Refresh Time (I/O Scan)**: set the refresh time of each scan of a Ladder (the minimum value is 100 ms)

Users can manage their ladder design in the Workspace.

5.3.1. Getting Started

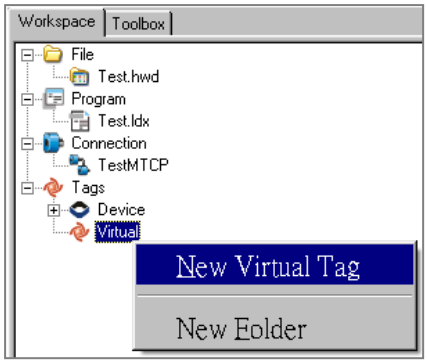
- To use **Ladder Designer**, run HMIWorks_Standard.exe first.

- Then create a new project.



- **New Virtual Tag** and open **Ladder Designer** from **HMI** menu

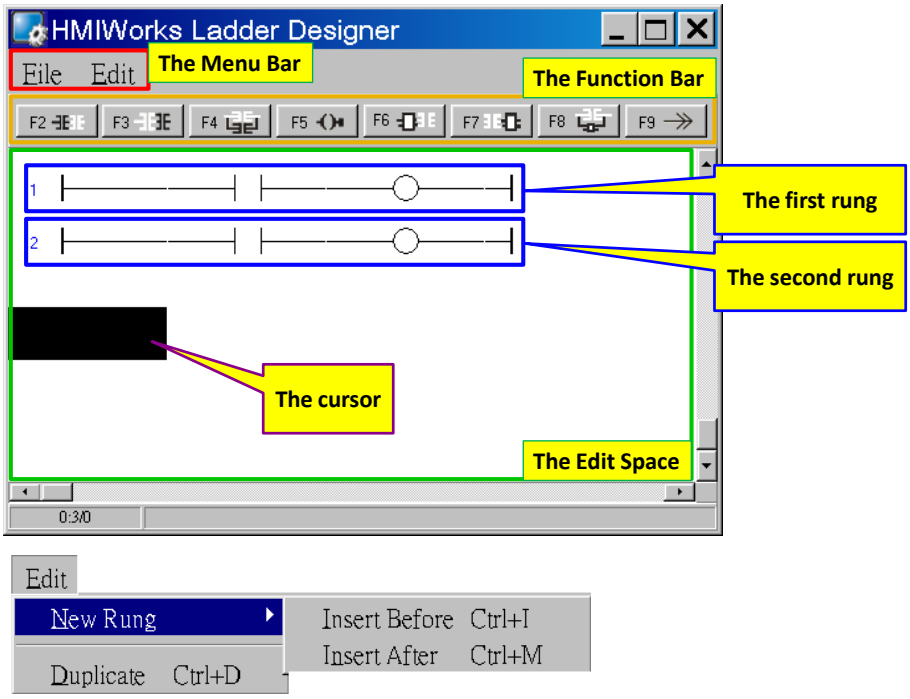
- **New Virtual Tag** adds variables used in **Ladder Designer**.
  There are three ways to open the "**Edit variable**" window,
  - pressing **F2** key,
  - choosing in **HMI** menu,
  - right-click on the **Virtual** item and click on "**New Virtual Tag**".



5.3.2. Introduction to Ladder Designer

A **Ladder Designer** is a tool to implement the ladder logic according to users' design.

Press **F4** to open **Ladder Designer**.



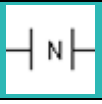
Mainly, a **Ladder Designer** consists of three parts, the menu bar, the function bar, and the edit space. The highlighted rectangle area is the cursor.

The briefings of the function bar:

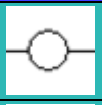
Item	Description

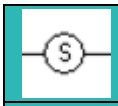
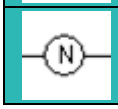
F2	Insert a contact input in the left of the cursor
F3	Insert a contact input in the right of the cursor
F4	Insert a contact input which is parallel to the cursor
F5	Insert a coil output
F6	Insert a function block in the left of the cursor
F7	Insert a function block in the right of the cursor
F8	Insert a function block which is parallel to the cursor
F9	Insert a Jump which is parallel to the cursor
Space [T]	Change the type of the contact input/ coil output

The briefings of the contact input type:

Item	Description
	A normally-open contact input
	A normally-closed contact input
	A positive transition contact input ➢ when the state from OFF to ON, trigger one shot
	A negative transition contact input ➢ when the state from ON to OFF, trigger one shot

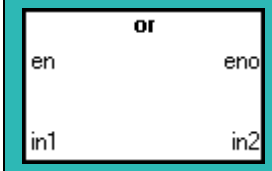
The briefings of the coil output type:

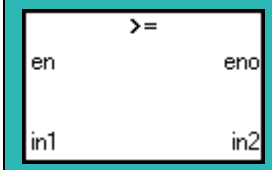
Item	Description
	A normally-open coil output
	A normally-closed coil output

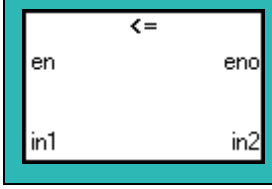
	A “Set” coil output ➢ once triggered, the coil remains ON until a reset
	A “Reset” coil output ➢ once triggered, the coil remains OFF until a set
	A positive transition coil output ➢ when the state from OFF to ON, trigger one shot
	A negative transition coil output ➢ when the state from ON to OFF, trigger one shot

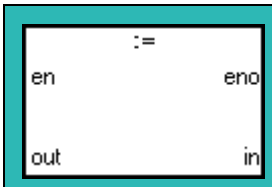
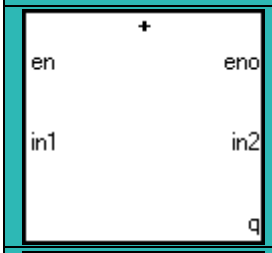
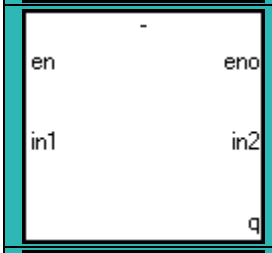
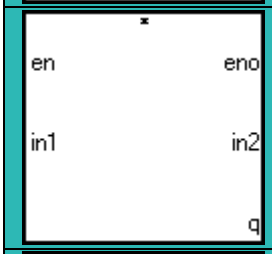
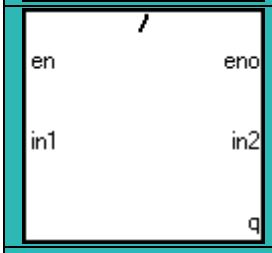
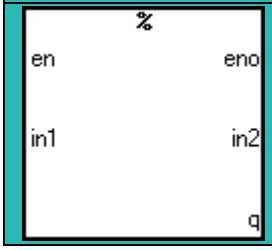
The briefings of function blocks

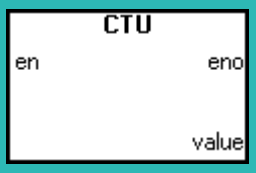
Refer to “C:\ICPDAS\HMIWorks_Standard\bin\FunctionBlock” for more details.

Item	Description (parts of C code)	Group
	**AND (And)** If en == 1, eno = in1 & in2; Else eno = 0.	default
	**OR (Or)** If en == 1, eno = in1 | in2; Else eno = 0.	default
	**XOR (Exclusive Or)** If en == 1, eno = in1 ^ in2; Else eno = 0.	default
	**NE (Not Equal)** If (en == 1 and in1 is not equal to in2), eno = 1; Else eno = 0;	default

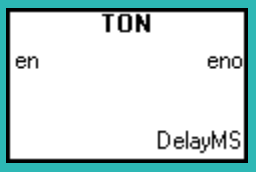
	**GE (Greater or Equal)**
	If (en == 1 and in1 >= in2),
	eno = 1;
	Else eno = 0;
		default

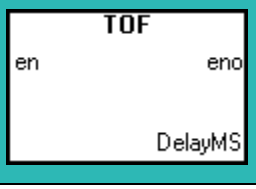
	**LE (Less or Equal)**
	If (en == 1 and in1 <= in2),
	eno = 1;
	Else eno = 0;
		default

	**Assign** If en == 1, “out” is assigned with “in” and eno = 1; Else eno = 0;	default
	**Add (Addition)** If en == 1, q = in1 + in2 and eno = 1; Else eno = 0;	math
	**Sub (Subtraction)** If en == 1, q = in1 - in2 and eno = 1; Else eno = 0;	math
	**Mul (Multiplication)** If en == 1, q = in1 * in2 and eno = 1; Else eno = 0;	math
	**Div (Division)** If en == 1, q = in1 / in2 and eno = 1; Else eno = 0;	math
	**Inc (Increment)** If en == 1, increment “in” by 1; Else eno = 0;	math
	**Dec (decrement)** If en == 1, decrement “in” by 1; Else eno = 0;	math
	**Mod (Modulo)** If en == 1, q = in1 % in2 and eno = 1; Else eno = 0;	math

	**CTU (Count Up)**
	End: count >= value.
	If en == 1,
	Count up until End,
	During counting, eno = 0,
	When End, eno = 1;
	Else
	Reset count to 0,
	eno = 0;
	
	Note: the counting period depends on the number of rungs
		counter

	**CTD (Count Down)**
	End: Count <= 0.
	If en == 1,
	Count down until End,
	During counting, eno = 0,
	When End, eno = 1;
	Else
	Reset count to value,
	eno = 0;
	
	Note: the counting period depends on the number of rungs
		counter

	**TON (Timer On, unit=ms)**
	End: elapsed >= DelayMS.
	If en == 1,
	Start the timer if not,
	Stop the timer when End,
	When timer runs, eno = 0,
	When End, eno = 1;
	Else
	Reset the timer,
	eno = 0;
		timer

	**TOF (Timer Off, unit=ms)** End: elapsed >= DelayMS. If en == 1, Start the timer if not,	timer

	Stop the timer when End,
	When timer runs, eno = 1,
	When End, eno = 0;
	Else
	Reset the timer,
	eno = 0;	

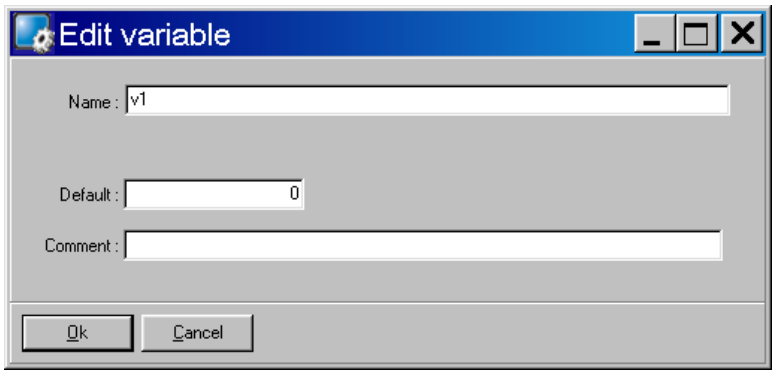
Beep en eno	**Beep** If en == 1, beep and eno = 1; Else eno = 0;	system

5.3.3. Operations of Ladder Designer

5.3.3.1. New Virtual Tags (F2)

To use **Ladder Designer**, add variables for **Ladder Designer** first.

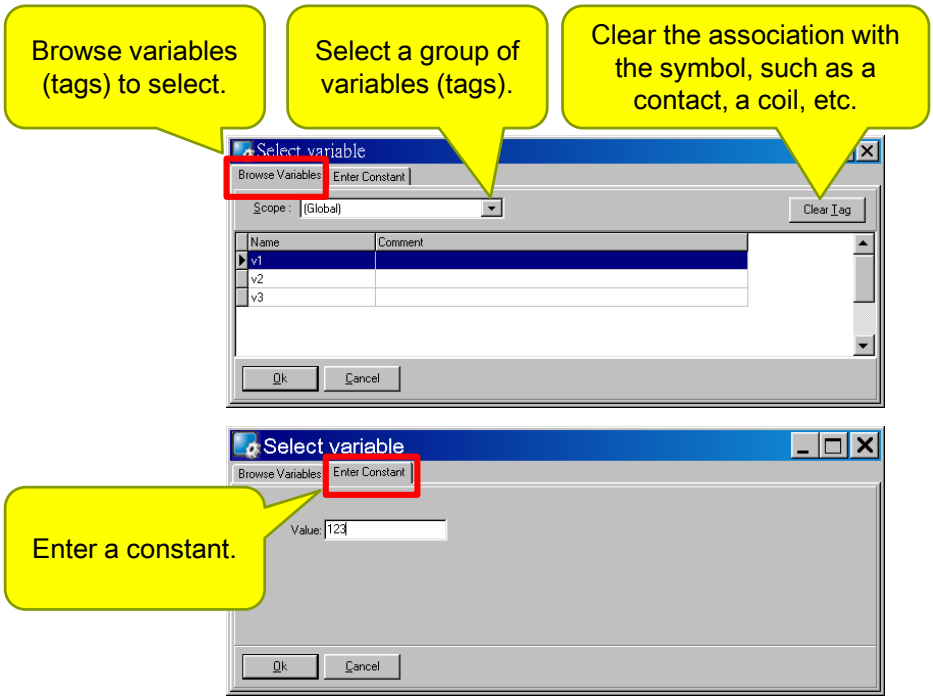
1. Press **F2 (HMI → New Virtual Tag)** to add virtual tags, an "**Edit variable**" window shows.
2. Define a new variable in the name field and optionally fill the other fields.
3. Finally, press **OK** button to take effect.



Here, we add three variable v1, v2 and v3 for example in the next sections.

5.3.3.2. Select Variables and Enter Constants

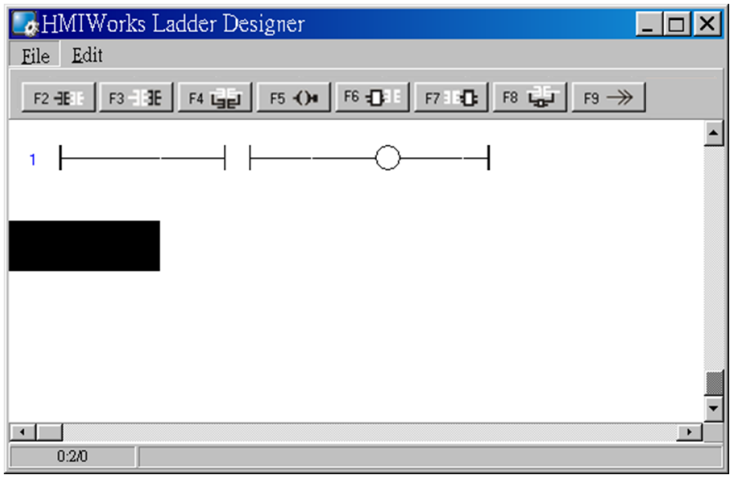
Double click on the symbol of contact inputs, coil outputs, etc. to evoke the "**Select variable**" window to select variables or enter constants as below.



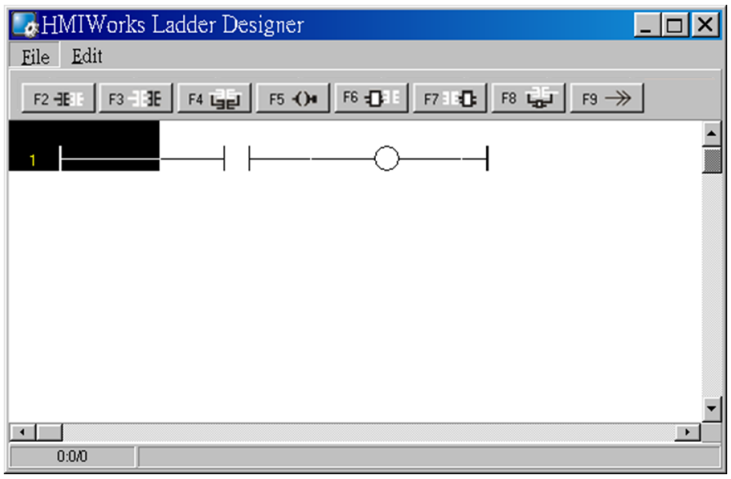
5.3.3.3. Insert/Delete a Rung

To insert a rung, move the cursor to the empty place and then press **F2** (or **F3**/**F4**).

(Or press **F6**, **F7**, **F8** to insert a rung with a function block.)

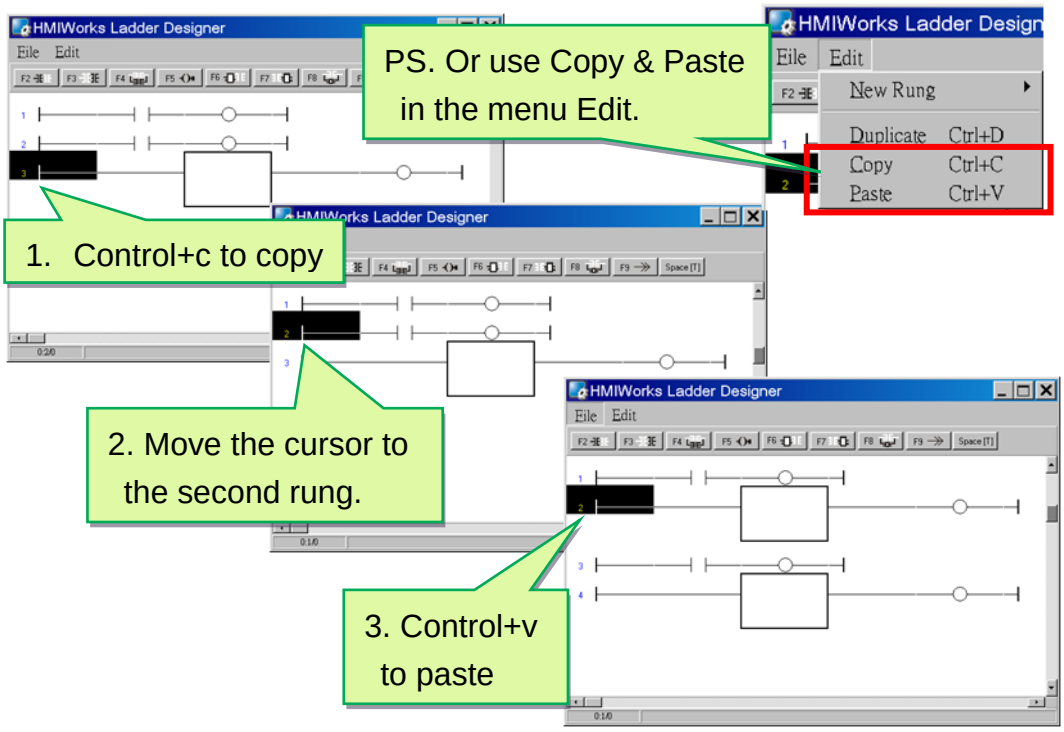


To delete a rung, move the cursor (the highlighted area) to the starting point of the rung and then press "**Delete**" key.



5.3.3.4. Copy and Paste a Rung

Supposed that we have three rungs and we want to copy the third rung and insert it between the first and the second rungs.



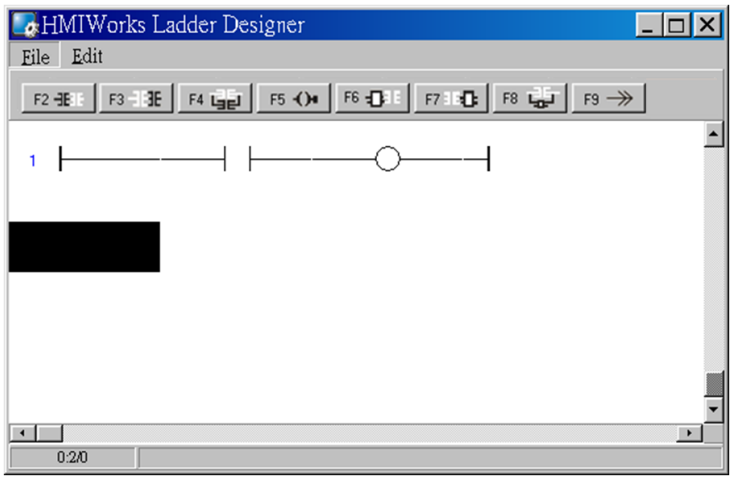
5.3.3.5. Insert/Delete a Contact Input in a Rung

To demonstrate how to insert/delete a contact input and other related issues,

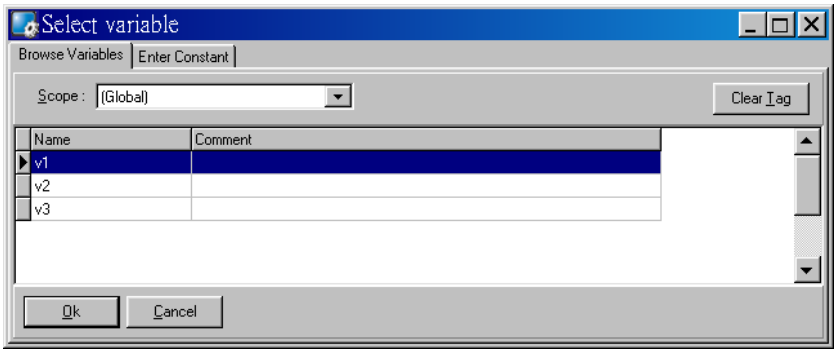
we embed each function in the sequence of steps as followings.

1. Associate a variable to a contact input

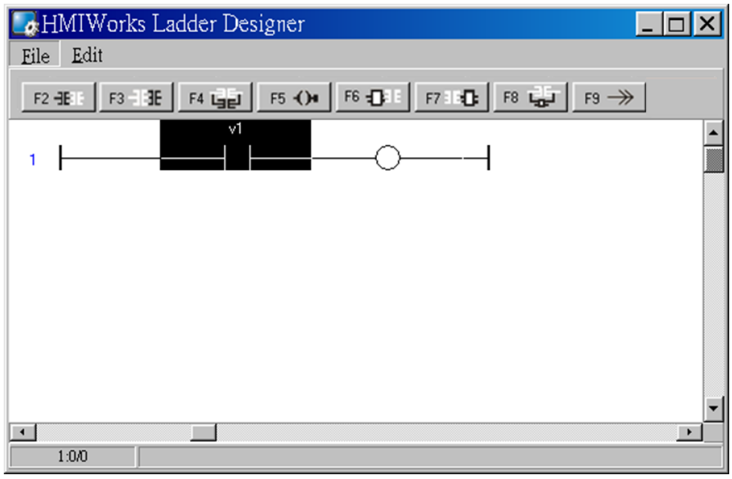
Press **F2** to insert a new rung with a contact input and a coil output.



In the new rung, double-click on the contact input to invoke the "**Select variable**" window to select a variable (tag) and set it to the contact input.



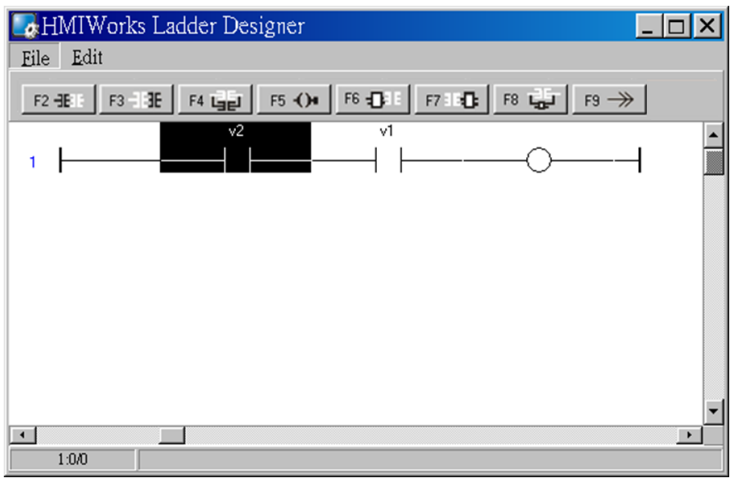
For example, we double-click on the variable "v1" and set to the contact input. v1, v2 and v3 are the variables set by "**New Virtual Tags**". Refer to the "**New Virtual Tags**" section.



2. Insert a new contact input in the left of the cursor (**F2**)

Move the cursor to the “v1” contact input and then press **F2**.

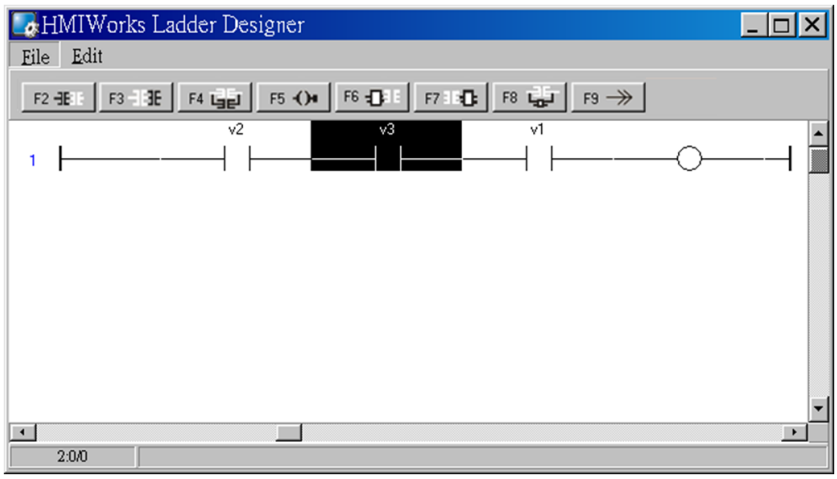
And to make things clear, associate variable “v2” to the newly-inserted contact input.



3. Insert a new contact input in the right of the cursor (**F3**)

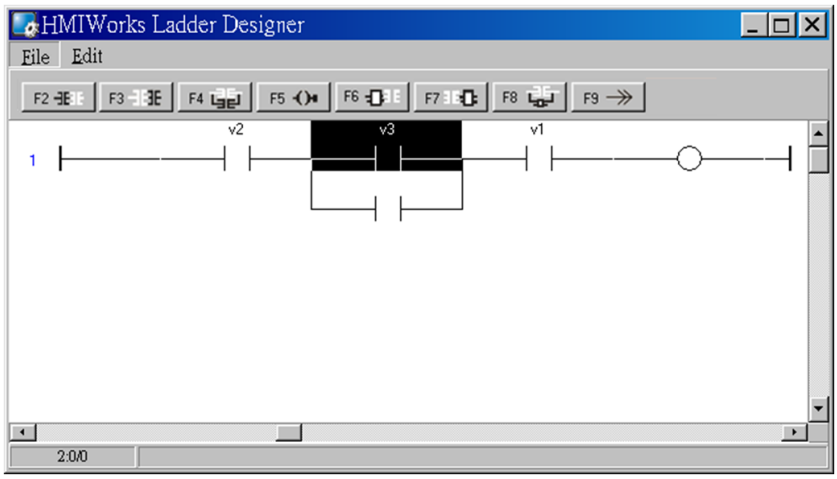
Move the cursor to the “v2” contact input and then press **F3**.

Associate variable “v3” to the newly-inserted contact input.



4. Insert a new contact input which is parallel to the cursor (**F4**)

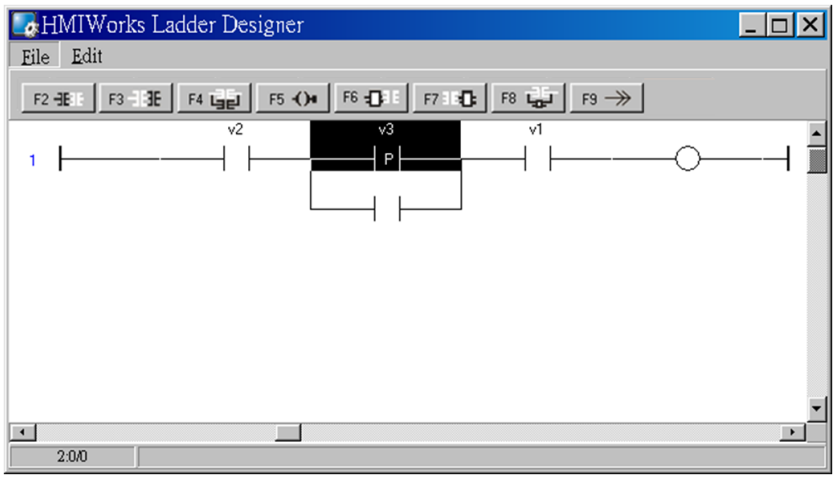
Move the cursor to the “v3” contact input and then press **F4**.



5. Set the type of a contact input

Move the cursor to a contact input and then press the space bar to change the type of the contact input.

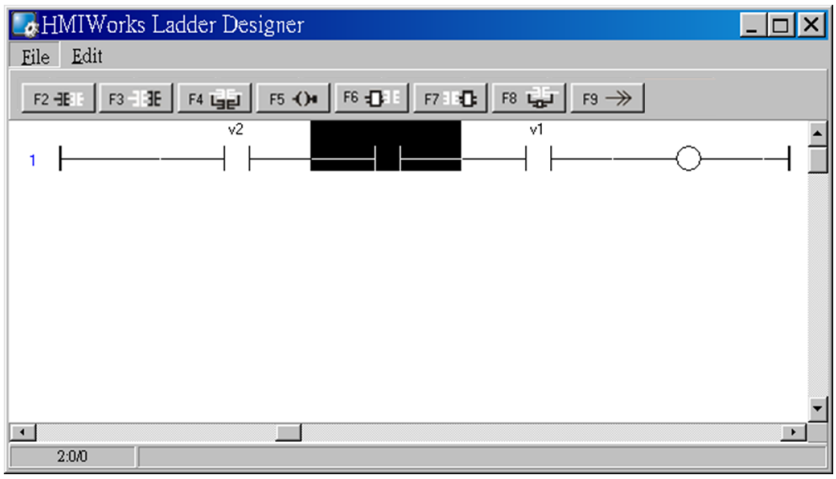
For example, we move the cursor to the “v3” contact input. Press the space bar twice to set the type of the contact input to pulse contact input.



6. Delete a contact input in the rung

Move the cursor to the contact input you want to delete. Then press the "**delete**" key.

For example, we move the cursor to the "v3" contact input and then press the "**delete**" key.

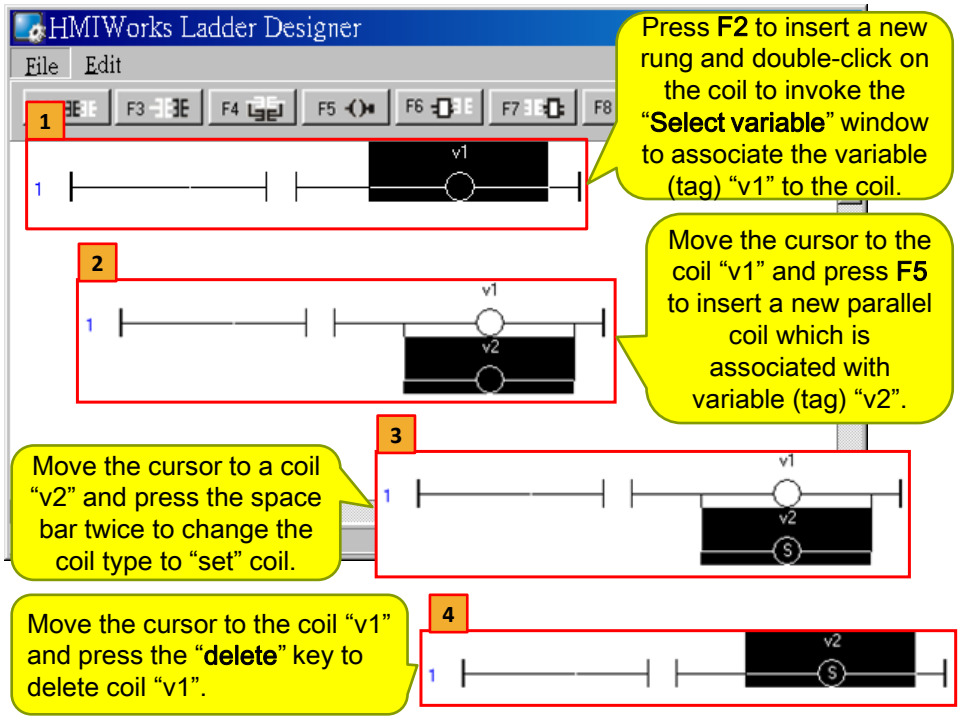


7. Delete the rung.

Move the cursor to the starting point of the rung and then press "**Delete**" key.

5.3.3.6. Insert/Delete a Coil Output in a Rung

To demonstrate how to insert/delete a coil output and other related issues, we embed each function in the sequence of steps as followings.



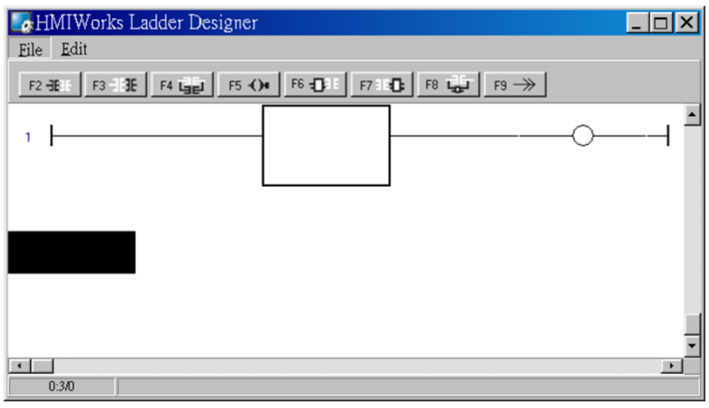
5.3.3.7. Insert/Delete a Function Block in a Rung

To demonstrate how to insert/delete a function block and other related issues, we embed each function in the sequence of steps as followings.

1. Set a function type to a function block

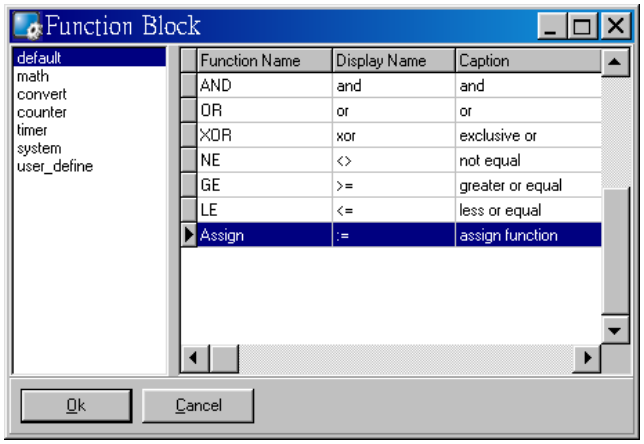
   i. Insert a new rung

   Press **F6** to insert a new rung with a function block and a coil output.



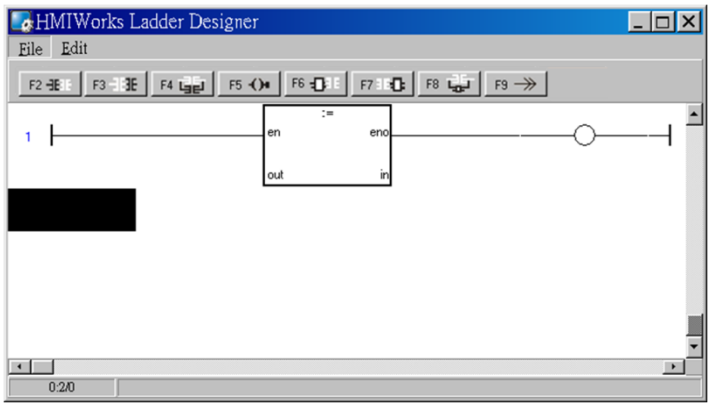
   ii. Choose function type

   In the new rung, double-click on the function block to invoke the "**Function Block**" window.



Double-click on the “Function Name” in the list to set the type of the function.

For example, we double-click on the Function “Assign” in the default group and set to the function block.



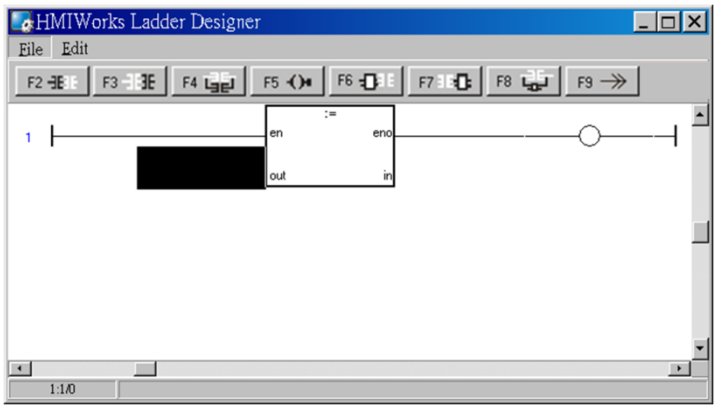
iii. Set the variables of the function

Now, we should set the variable to the function “Assign”. As you can see, there are four variables, en, eno, out, in.

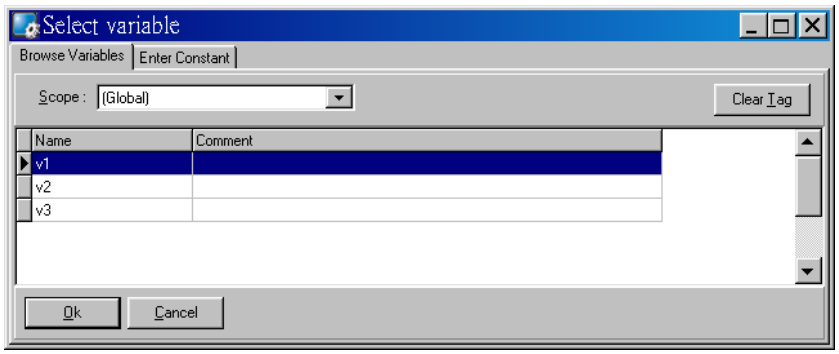
- Both “en” and “eno” cannot associate variables by users.
- We can associate “out” and “in” with the variables we define by “**New Virtual Tags**”.

For example, we associate “v1” to “out” and “v2” to “in”. v1, v2 and v3 are the variables set by “**New Virtual Tags**”. Refer to the “**New Virtual Tags**” section.

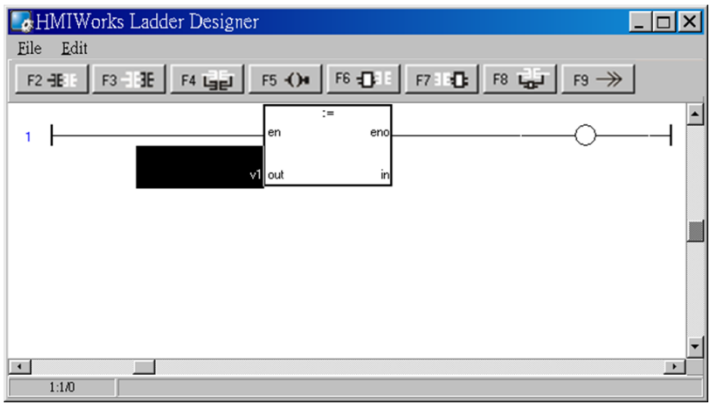
To associate “v1” to “out”, move the cursor just beside “out” but not in the function block.



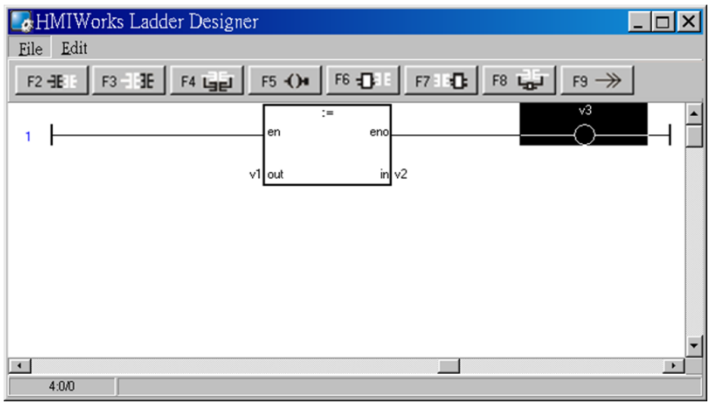
Double-click on **just beside "out"** to invoke "**Select variable**" window.



Double-click on the variable in the list to set the variable to "out". For example, we double-click on the variable "v1" and set to "out" of "Assign" function.



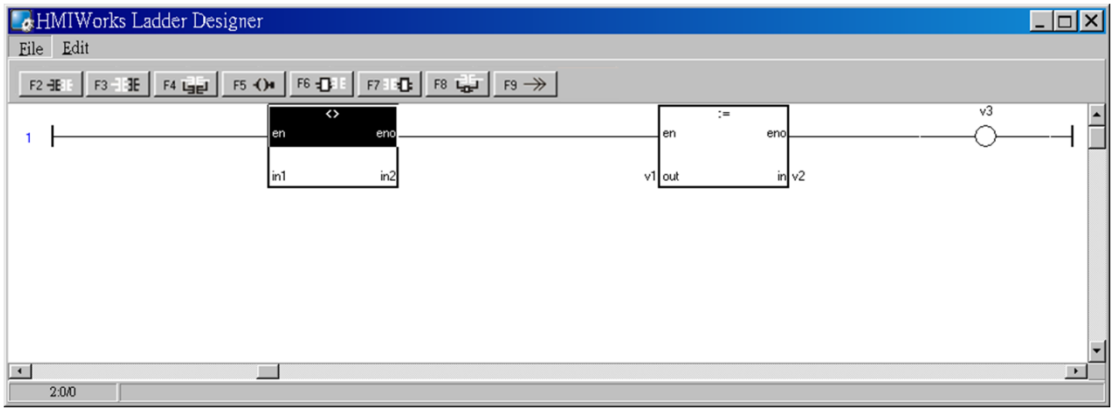
Set "v2" to "in" of "Assign" function in the same way.
Finally, set "v3" to the coil output.



This function assigns “v2” to “v1” if en is set to high.
The coil output “v3” is purely defined by eno, where eno = en.

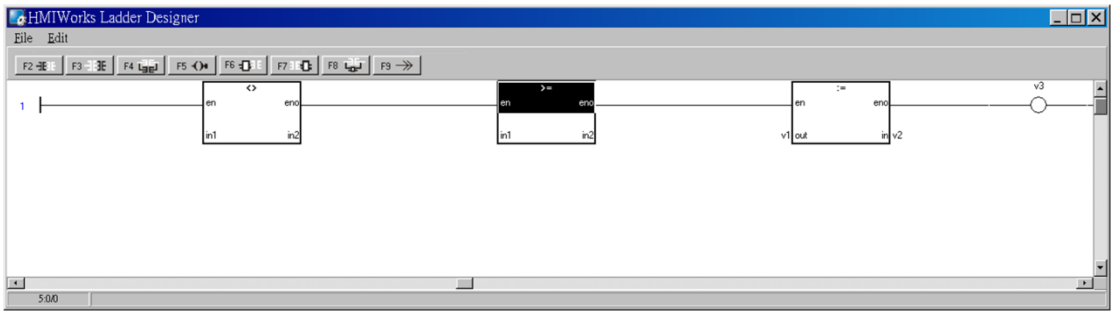
2. Insert a new function block in the left of the cursor (**F6**)

Move the cursor to the “Assign” function block and then press **F6**.
And to make things clear, set the newly-inserted function block as “NE” (not equal).

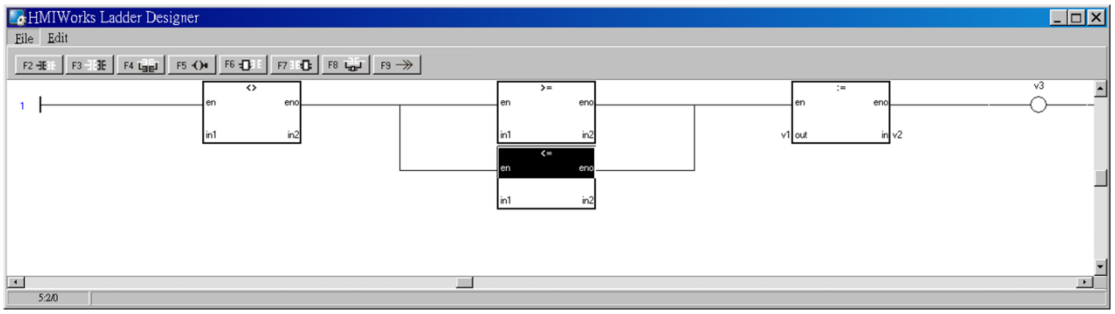


3. Insert a new function block in the right of the cursor (**F7**)

Move the cursor to the “NE” function block and then press **F7**.
Set the newly-inserted function block as “GE” (greater or equal).



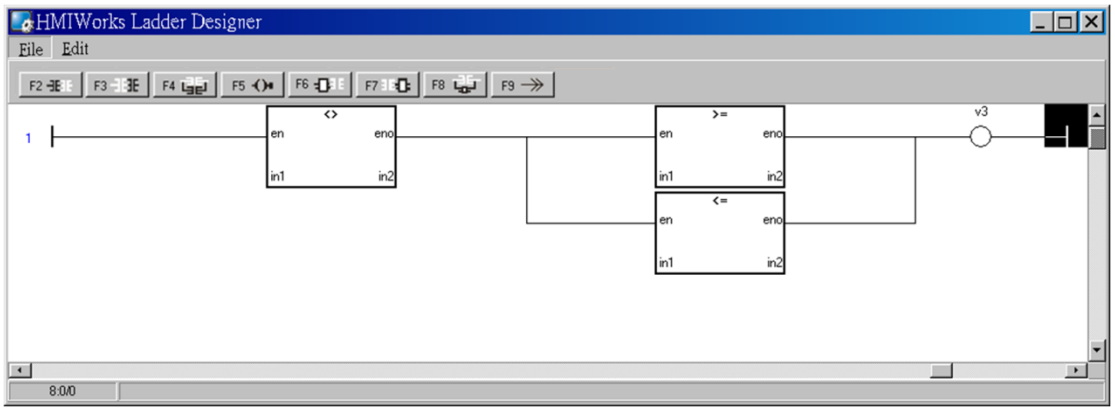
4. Insert a new function block which is parallel to the cursor (**F8**)
   Move the cursor to the “GE” function block and then press **F8**.
   Set the newly-inserted function block as “LE” (less or equal).



5. Delete a function block in the rung

Move the cursor to the function block you want to delete. Then press the “**delete**” key.

For example, we move the cursor to the “Assign” function block and then press the “**delete**” key.



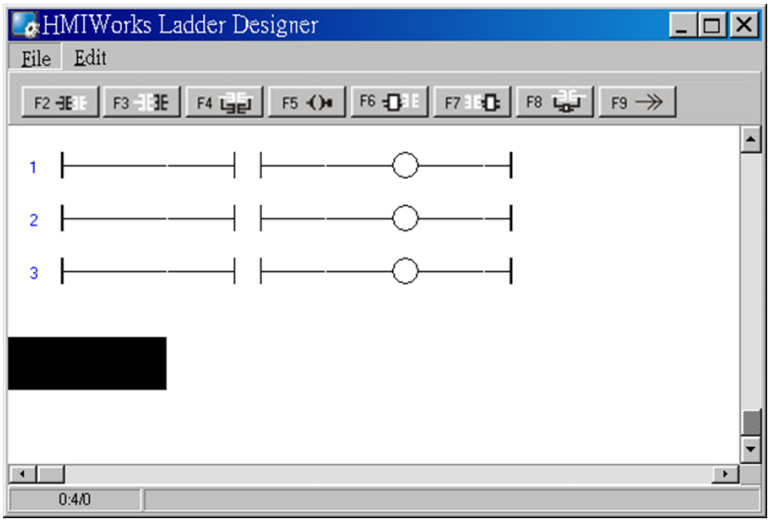
6. Delete the rung.

Move the cursor to the starting point of the rung and then press “**Delete**” key.

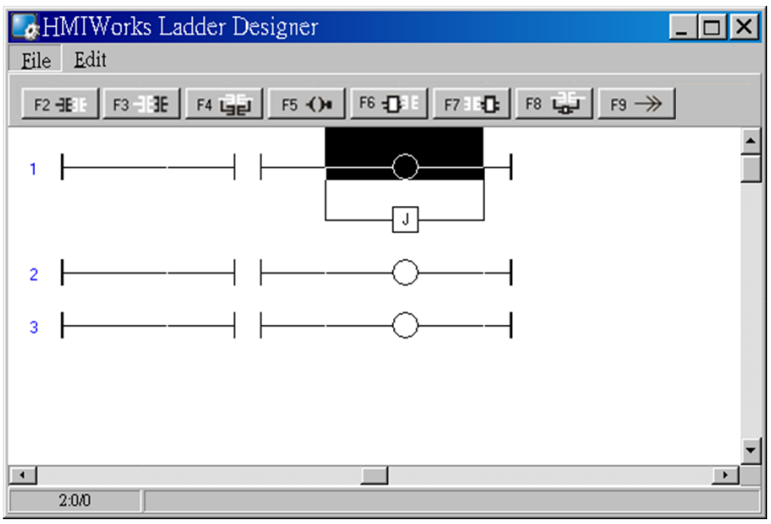
5.3.3.8. Jump to a Label

To demonstrate how to jump to a label, we first create three rungs and then explain how to skip the second rung and jump to the third.

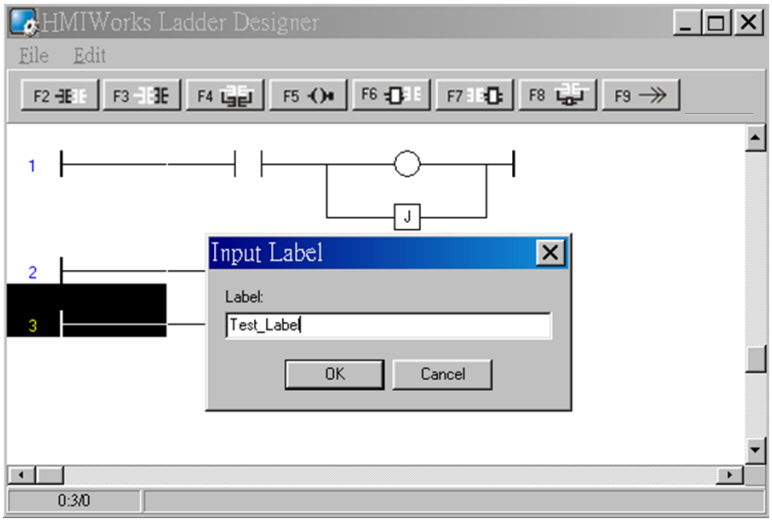
1. Press **F2** three times to create three rungs for example.



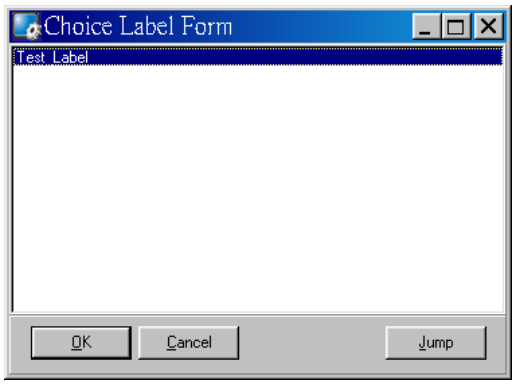
2. Move the cursor to the coil output of the first rung and then press **F9** to add a Jump.

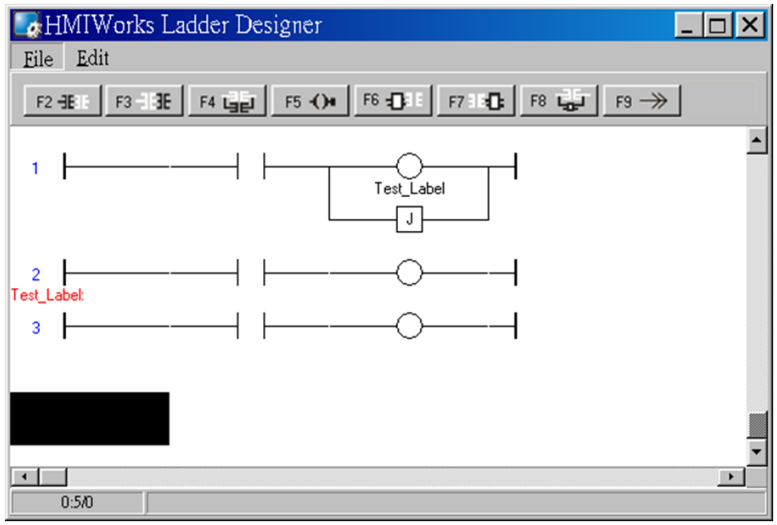


3. Double click on the starting point of the third rung to add a label “Test_Label” to it.



4. Double click on the Jump of the first rung to associate with the label of the third rung.





5. When running the ladder logic, set the coil output of the first rung to high, skip the second rung and jump to the third rung if the contact input of the first rung is closed.

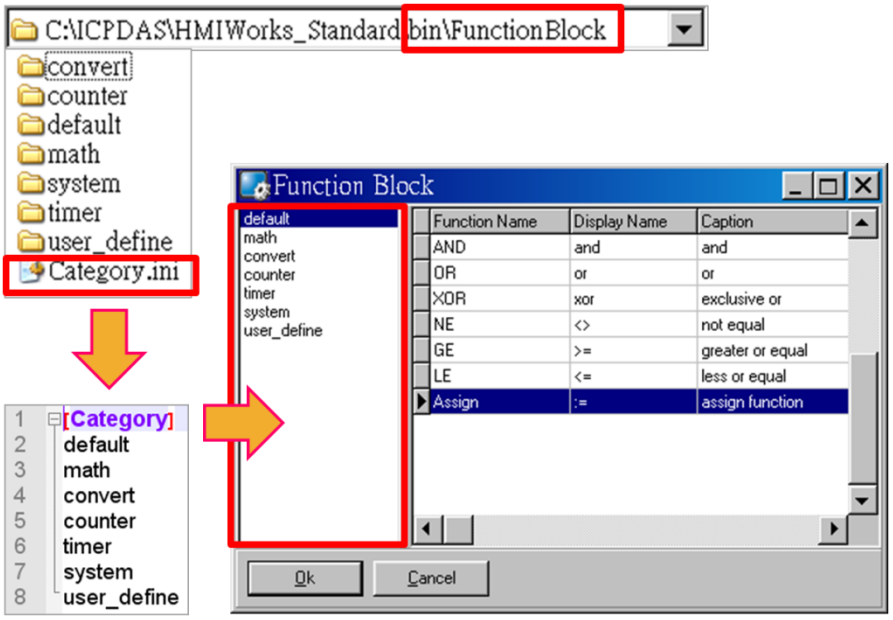
5.3.4. User-Defined Function Block

Why should we use function block?
There may be cases that using only ladders is too complex. At that time, function block may be a good choice.

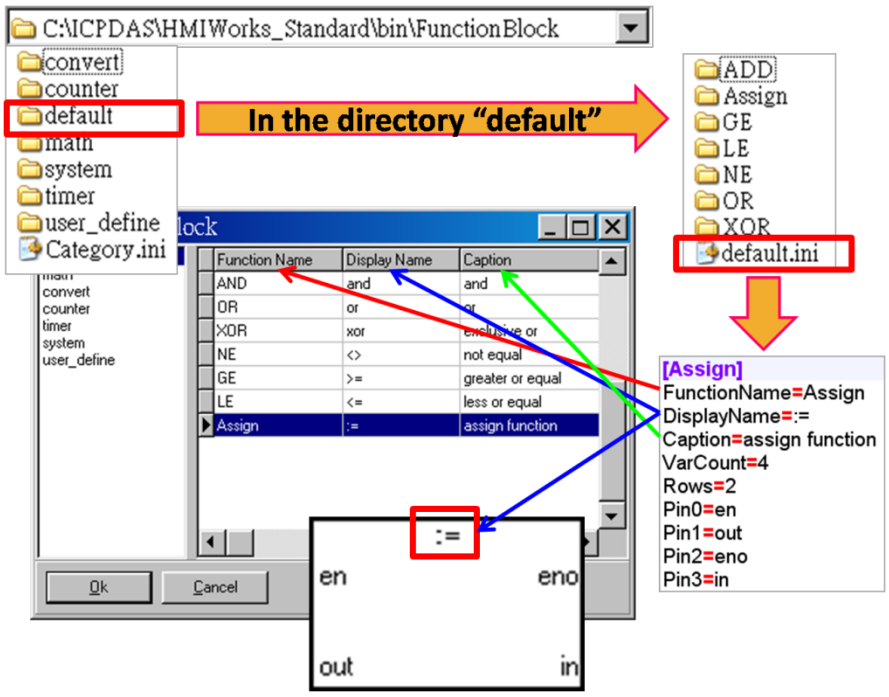
To know how to add a user-defined function block, we first explain how HMIWorks uses these function blocks. Take function block “Assign” in the group “default” for example.

How HMIWorks Uses Function Blocks

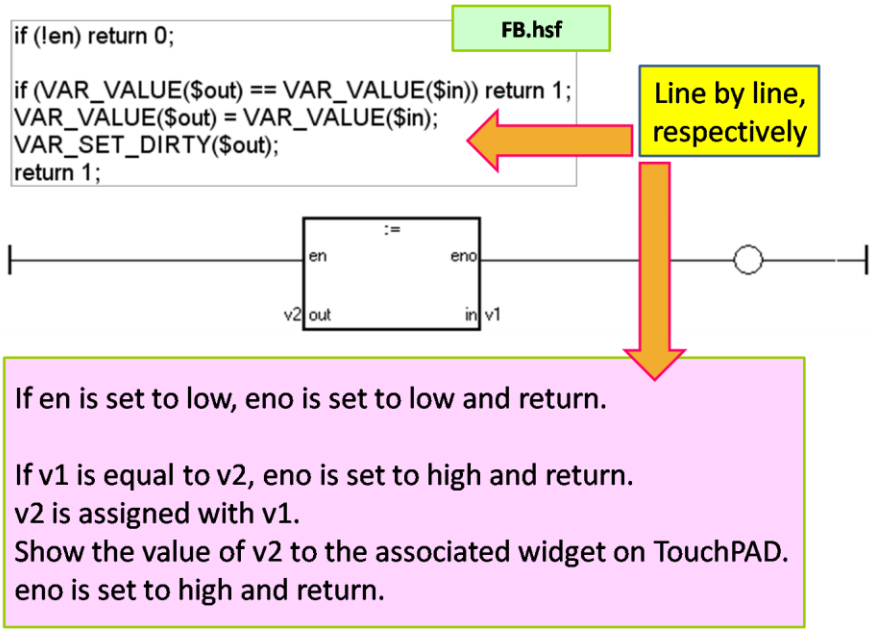
1. Go to the installation path of HMIWorks. In the sub-directory of “bin\FunctionBlock”, open the file “Category.ini” to load the groups.



2. If we choose group “default”, HMIWorks open and load from the .ini file in the sub-directory both of the same name “default”. That is, the “default.ini” in the sub-directory “default”.



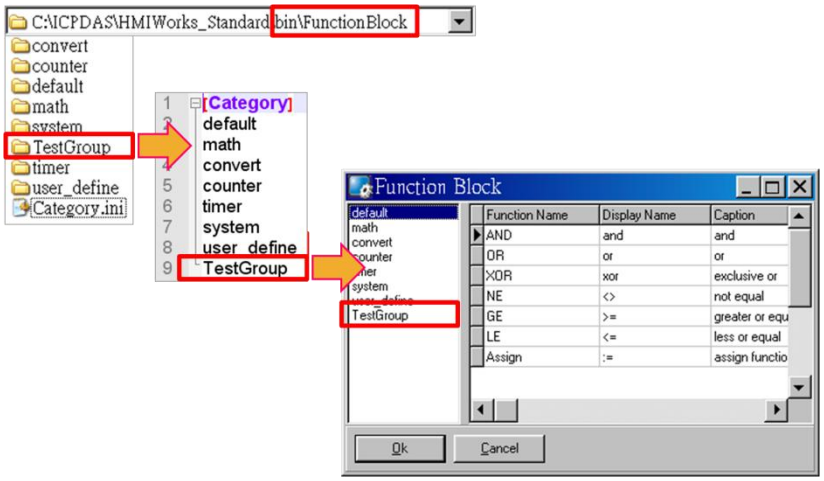
3. Double click on the "Assign" to use in the **Ladder Designer**. **Ladder Designer** uses the logics defined in the file FB.hsf in the sub-directory "Assign". FB.hsf uses C language. Following figure explains what FB.hsf of "Assign" does.



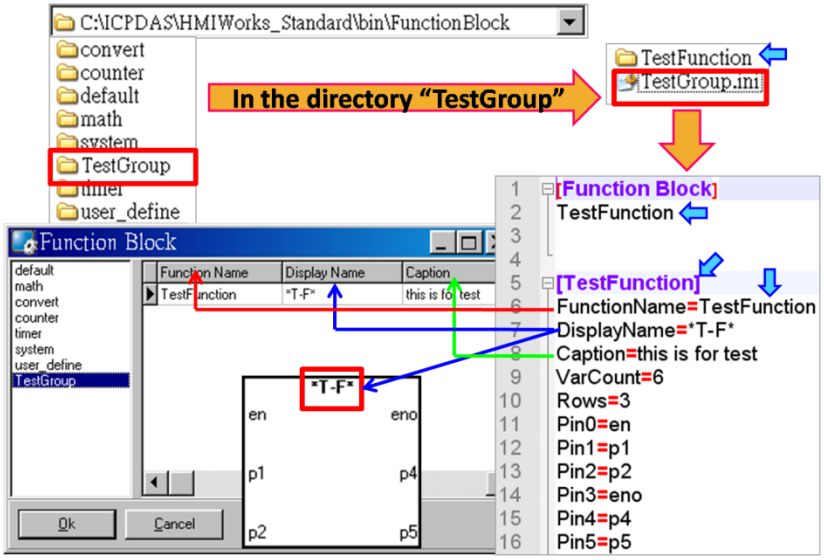
How to Add a User-Defined Function Block

1. Create a new group if necessary.

Go to the installation path of HMIWorks. In the sub-directory of “bin\FunctionBlock”, create a new directory “TestGroup” for example and open the file “Category.ini” to add a new item to represent the new group. **Note** that: the name of the new item in the Category.ini **must** be exactly the same as the name of the newly-created directory.

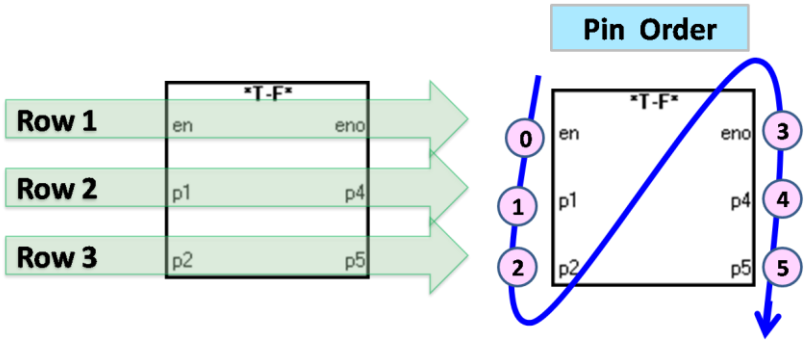


2. Go to the directory “TestGroup”, create a .ini file of the exactly same name as that of the group, that is, “TestGroup”. Create a sub-directory of the TestGroup and we may call the sub-directory “TestFunction”. Finally, define newly-defined function, “TestFunction” in the file “TestGroup.ini”.



Note that: VarCount = pin counts.

Below shows what does the Row mean and the order of the pins.



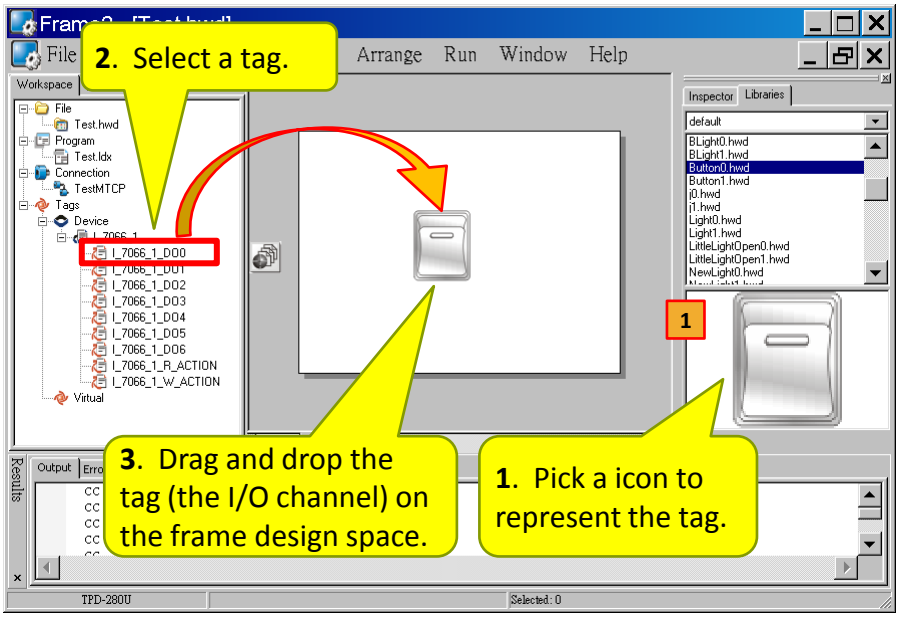
3. In the directory “TestFunction”, create a new file FB.hsf to implement the user-defined function.

5.3.5. Associate Tags with Tools

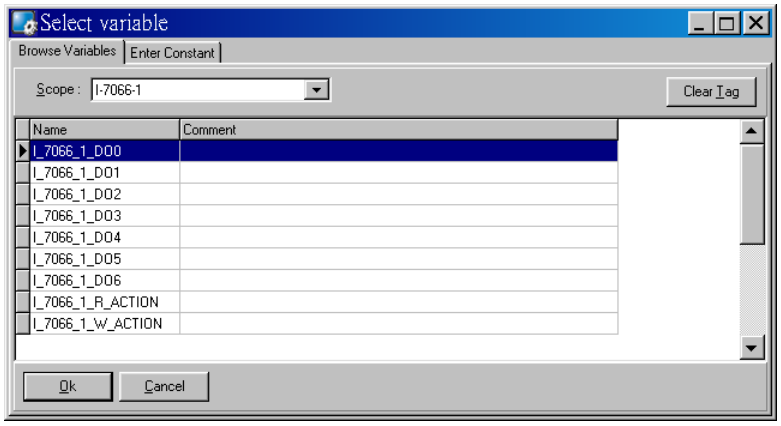
In order to use **Ladder Designer** to build HMI of TouchPAD, we should associate tags with tools. Until now, we can use Slider, CheckBox, and Label to associate themselves with tags.

There are three methods to associate tools with tags. Every change of the tag in the **Ladder Designer** is updated to the tool in the run time after association.

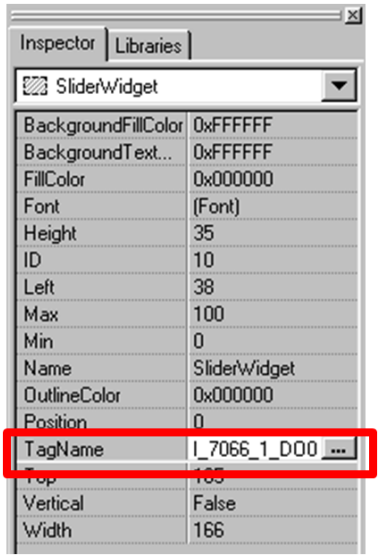
1. The first method: simply drag and drop the tags in the **Workspace** to the frame design space. A CheckBox is created with the tag associated.
   Note: this feature is only supported for CheckBox.



2. The second method: double click on the widget on the frame design space to evoke the "**Select variable**" window. Take a Slider for example.



Double click on the tag Name you want to associate with the widget. Then you can see the tag is associated with the widget (that is, the Slider for example) by setting the property TagName to the name of the tag.



3. The third method is click on the "**…**" button of the TagName property in the inspector of the Widget to evoke the "**Select variable**" window. Similar steps as above.

Special Note:
Refer to section "Using an ObjectList". Set RefObject property of a CheckBox to an ObjectList which contains images and then associate a tag of the **Ladder Designer** to the CheckBox. Then every time the tag changes its value, the CheckBox toggles the images. This feature is especially useful when building switches.

5.3.6. User-Defined I/O Modules

To know how to add a user-defined I/O module, we first explain how HMIWorks uses these I/O modules.

There are several kinds of I/O modules.

- DCON I/O modules: I-7000 series I/O modules by ICP DAS.

http://www.icpdas.com/products/Remote_IO/i-7000/i-7000_introduction.htm

- Modbus TCP I/O modules: ICP DAS provides ET/PET-7000 series.

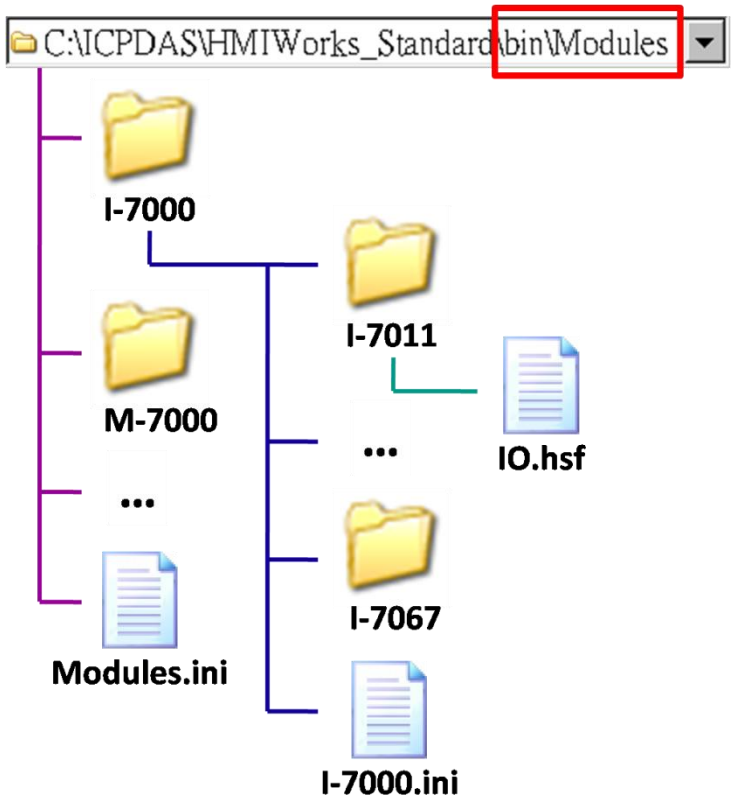
http://www.icpdas.com/products/Remote_IO/et-7000/et-7000_introduction.htm

- Modbus RTU I/O modules: M-7000 series I/O modules by ICP DAS

http://www.icpdas.com/products/Remote_IO/m-7000/m-7000_introduction.htm

Where HMIWorks Put I/O Module Information

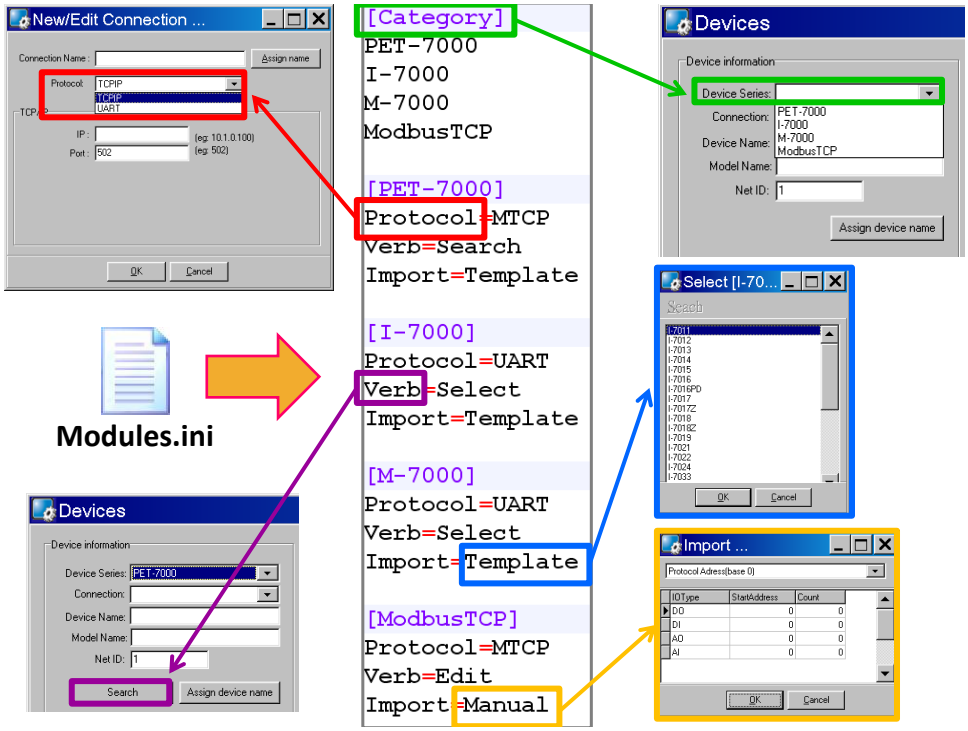
HMIWorks puts I/O module information in the following locations.



Some explanations for above figure:

- "C:\ICPDAS\HMIWorks_Standard\" is the default installation path. (Users may have different installation paths.)
- Modules.ini is the I/O series configuration file.
- I-7000.ini is the I/O modules configuration file for I-7000 series. M-7000.ini and PET-7000.ini are I/O modules configuration file for M-7000 and PET-7000 series respectively.
- Each I/O module has a matching name directory and in that directory there is only one file, IO.hsf. IO.hsf is the file of C language to define the behaviors of the I/O module.
- I/O module directories in the same series are grouped together in the I/O series directory. For example, I-7011, …, I-7067 are directories represent I/O modules and they are all put to the series directory "I-7000".

What Module.ini describes?



In details, we have the following table:

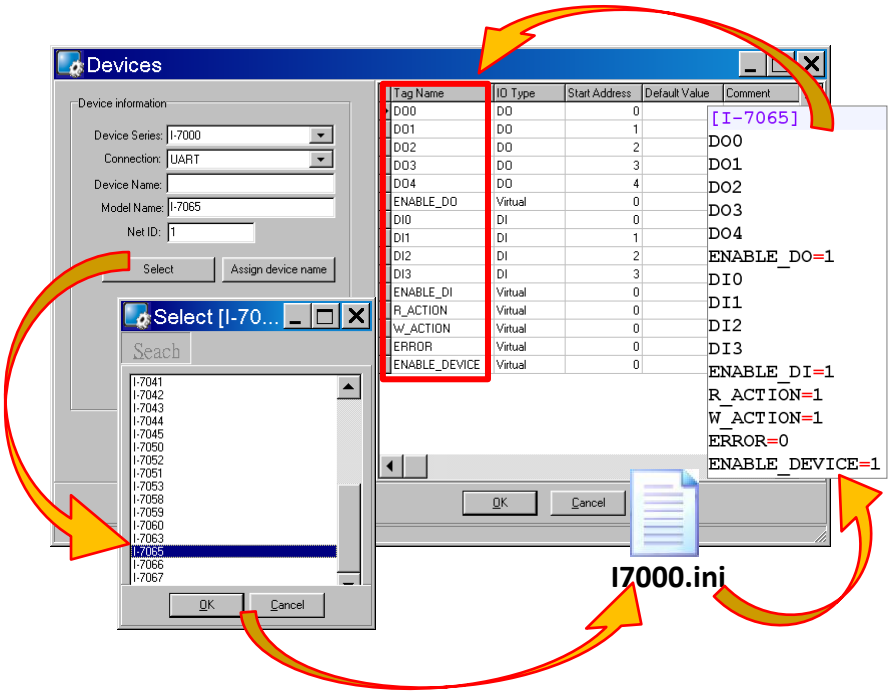
Item		Description
Category		This section keeps the list of the device series which HMIWorks supports. When registering device (**F3**), the "**Devices**" window gets the information of device series from this "Category" section.
Protocol	MTCP	"Protocol=MTCP" in the Module.ini is corresponding to "Protocol=TCPIP" in the "**New/Edit Connection**" window in the **Workspace**.
	UART	"Protocol=UART" in the Module.ini is corresponding to "Protocol=UART" in the "**New/Edit Connection**" window in the **Workspace**.
Verb	Search	HMIWorks scans through the network to find out I/O modules. Until now, PET-7000 is the only series which support this "Search" function.

	Select	HMIWorks pops up a list of I/O modules to let users select one. The list of I/O modules is loaded from the file whose name is [Device_Series_Name].ini
	Edit	HMIWorks evokes the "**Import**" window to let users decide the I/O points for the I/O module.
Import	Template	HMIWorks imports the tags of the I/O module from the I/O module configuration file. For example, HMIWorks imports tags of I-7011 from the template in the file of I-7000.ini.
	Manual	HMIWorks imports the tags of the I/O module by the manually-decided I/O points.

How "Register Devices (F3)" Generates Tags

Press **F3** to evoke the "**Devices**" window to register I/O devices.

The I/O modules configuration file has templates for all the I/O modules in the I/O series. For example, I-7000.ini is the I/O modules configuration file for the I-7000 I/O series.
Take I-7065 in the I-7000 I/O series for example as the following figure shows.



As above, ERROR is the tag for the communication status.

How IO.hsf Defines I/O Behaviors

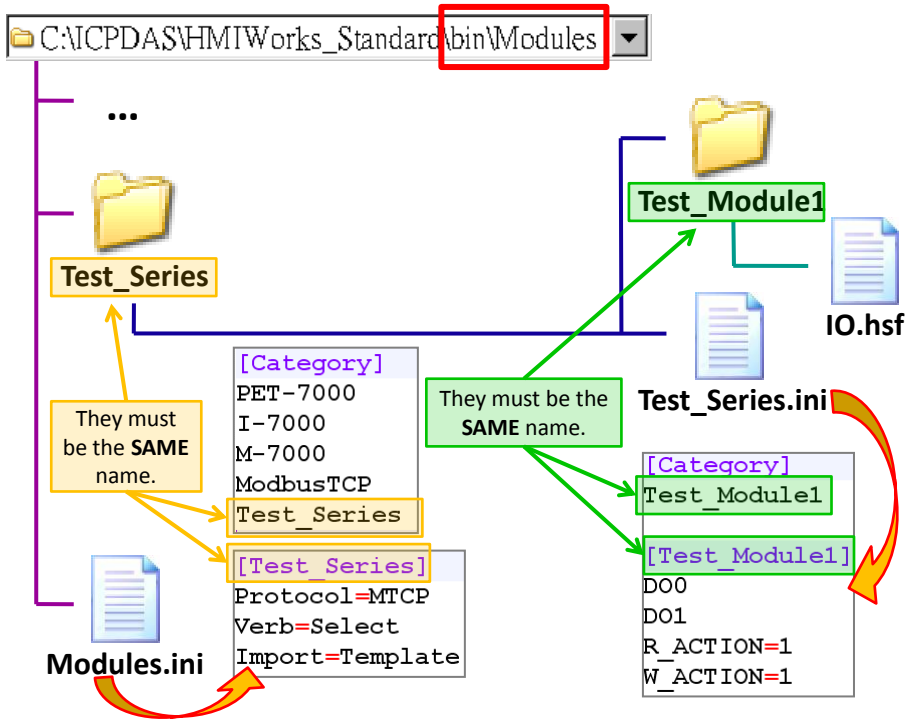
- Take I-7065 for example (I-7000 series I/O module)
  Open the IO.hsf in the directory “[HMIWorks install path]\bin\Modules\I-7000\I-7065\”.
  The codes in IO.hsf are of C language as below:

```
BEGIN_FUNCTION_BLOCK(); //must write

DWORD v_do = 0;
DWORD v_di = 0;
int     gWriteCount = 0;

//$W_ACTION: a tag used in Ladder to enable/disable writing actions
//$ENABLE_DO: a tag used in Ladder to enable/disable the part of DOs
if ( VAR_VALUE($ENABLE_DO) && VAR_VALUE($W_ACTION))
{
    int iWrite = 0; //To decide if there’s a need to write any DO channel
    v_do     = 0;
```

```
    if( VAR_FLAG_IS_WRITE($DO0) ) iWrite++;
    // VAR_FLAG_IS_WRITE: is there any change on this channel
    if(             VAR_VALUE($DO0) ) v_do |= (1<<0);
    // VAR_VALUE: used to get the value of this channel
    if( VAR_FLAG_IS_WRITE($DO1) ) iWrite++;
    if(             VAR_VALUE($DO1) ) v_do |= (1<<1);
    if( VAR_FLAG_IS_WRITE($DO2) ) iWrite++;
    if(             VAR_VALUE($DO2) ) v_do |= (1<<2);
    if( VAR_FLAG_IS_WRITE($DO3) ) iWrite++;
    if(             VAR_VALUE($DO3) ) v_do |= (1<<3);
    if( VAR_FLAG_IS_WRITE($DO4) ) iWrite++;
    if(             VAR_VALUE($DO4) ) v_do |= (1<<4);

    if ( iWrite )    // Write only when need
    {
        gWriteCount++;
        if ( ! dcon_WriteDO($DEVICE, $NETID, 5, v_do & 0xFF) )
        // dcon_WriteDO: the DO writing API function of I-7000 I/O series.
        // I-7000 I/O series uses the DCON protocol.
            return HMI_ERROR;
    }
}

if ( gWriteCount ) return HMI_OK;
// Skip reading to reduce the device loading

if ( (VAR_VALUE($ENABLE_DO) || VAR_VALUE($ENABLE_DI)) &&
VAR_VALUE($R_ACTION)) {
//$R_ACTION: a tag used in Ladder to enable/disable reading actions
//$ENABLE_DO: a tag used in Ladder to enable/disable the part of DOs
//$ENABLE_DI: a tag used in Ladder to enable/disable the part of DIs
if (dcon_ReadDIO($DEVICE, $NETID, 4, 5, &v_di, &v_do))
// dcon_ReadDIO: the DI/DO reading API function of I-7000 I/O series.
// I-7000 I/O series uses the DCON protocol.
{
    VAR_SET($DI0, v_di & (1<<0));
    // VAR_SET: used to set the value of this channel to its tag
    VAR_SET($DI1, v_di & (1<<1));
```

```
        VAR_SET($DI2, v_di & (1<<2));
        VAR_SET($DI3, v_di & (1<<3));

        VAR_SET($DO0, v_do & (1<<0));
        VAR_SET($DO1, v_do & (1<<1));
        VAR_SET($DO2, v_do & (1<<2));
        VAR_SET($DO3, v_do & (1<<3));
        VAR_SET($DO4, v_do & (1<<4));
    } else
        return HMI_ERROR;
    }

    END_FUNCTION_BLOCK(); //must write
```

Create a User-Defined I/O Module



1. In the directory, “[HMIWorks install path]\bin\Modules\”, create a new I/O series directory whose name is “Test_Series” and be sure to update Modules.ini to notify HMIWorks that there is a new I/O series called “Test_Series”. As the figure shows, the series directory name and the name in the Modules.ini must be **the same**.

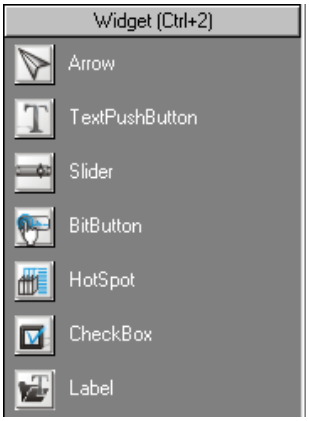
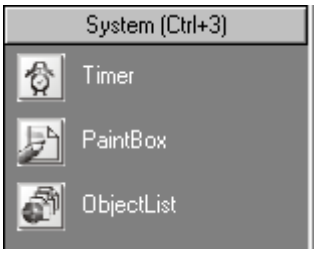
2. In the I/O series directory, “Test_Series”, we create a new I/O module directory whose name is “Test_Module1” and be sure to create a I/O modules configuration file, Test_Series.ini, to depict the template of the newly-created I/O module, Test_Module1. As the figure shows, the module directory name and the name in the Test_Series.ini must be **the same**.
3. Implement the IO.hsf which is created in I/O module directory, Test_Module1, to describe the behaviors of the I/O module, Test_Module1. Refer to the IO.hsf
   I. of PET-7000 series if using the Modbus TCP protocol.
   II. of M-7000 series if using the Modbus RTU protocol.
   III. of I-7000 series if using the DCON protocol.
   All are similar to the example of the I-7065 above.

5.4. Frames and Tools

This section introduces properties and usages of frames and tools.

In the **Toolbox**, there are three kinds of tools, Drawings, Widgets and System componets.

	**Drawing:** 1. **Rectangle**: draw a rectangle. 2. **Ellipse**: draw ellipse. 3. **Text**: put string (text) on screen. 4. **Picture**: load an image file on a frame. 5. **Line**: draw a line.

	**Widget**: 1. **TextPushButton**: create a button. 2. **Slider**: show or decide the percentage. 3. **BitButton**: create an image button. 4. **HotSpot**: create a hot spot that can issue an OnClick event. 5. **CheckBox**: provide an alternative. 6. **Label**: provide a string that can be modified during run-time.
	**System:** 1. **Timer**: periodically execute codes. 2. **PaintBox**: draw shapes in the run time. 3. **ObjectList**: maintain a list of library objects which can be used through property “RefObject” of TextPushButton and CheckBox.

Important Notice:

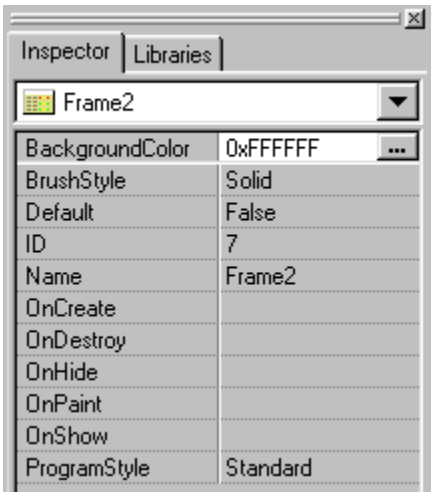
1. Make sure that widgets should not overlap or unexpected behavior may happen when clicking.
2. The touch area of a widget is the rectangle enclosed by the widget’s outline.

5.4.1. Properties of a Frame

This section introduces the properties of a frame.

Properties of a Frame

Click on the frame, and properties of the frame are shown in the **inspector**.

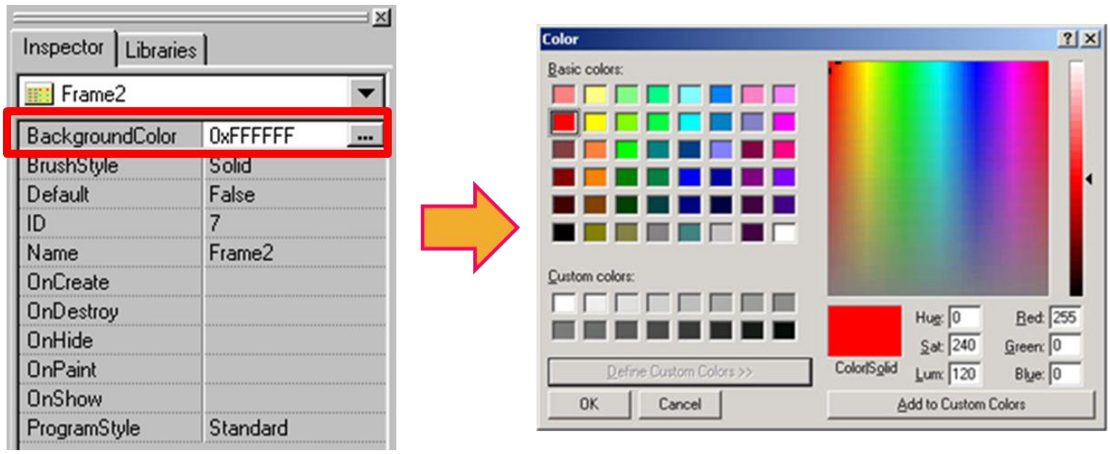


properties	description
BackgroundColor	The color of the background of the frame. The color is represented by a three byte value in hexadecimal form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
BrushStyle	Solid or Clear. If BrushStyle is set to Solid, BackgroundColor is effective. However this may make **screen flash** if background color is quite different from the loaded picture. Setting BrushStyle Clear disables the property BackgroundColor and prevents from flashing.
Default	Whether this frame is default frame or not. The default frame shows first after power on.
ID	The serial number of tools or frames which is used to identify them.
Name	The name of the frame
OnCreate	The function name of the OnCreate event of the frame. Use OnCreate to perform some operations when the frame is created.
OnDestroy	The function name of the OnDestroy event of the frame. Use OnDestroy to perform some operations when the frame is destroyed.
OnHide	The function name of the OnHide event of the frame. Use OnHide to perform some operations when the frame is hidden.
OnPaint	The function name of the OnPaint event of the frame. Use OnPaint to perform some operations

	when the frame is redrawn.
OnShow	The function name of the OnShow event of the frame. Use OnShow to perform some operations when the frame is shown.
ProgramStyle	Standard C or Ladder

How to change the BackgroundColor of a frame

Click on the BackgroundColor in the **inspector**. Then click on the button "**…**" to open color window to select color.

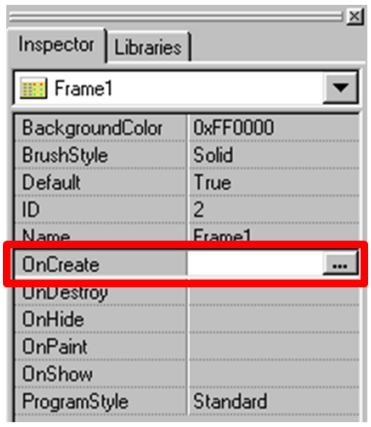


How to implement event handlers of a frame

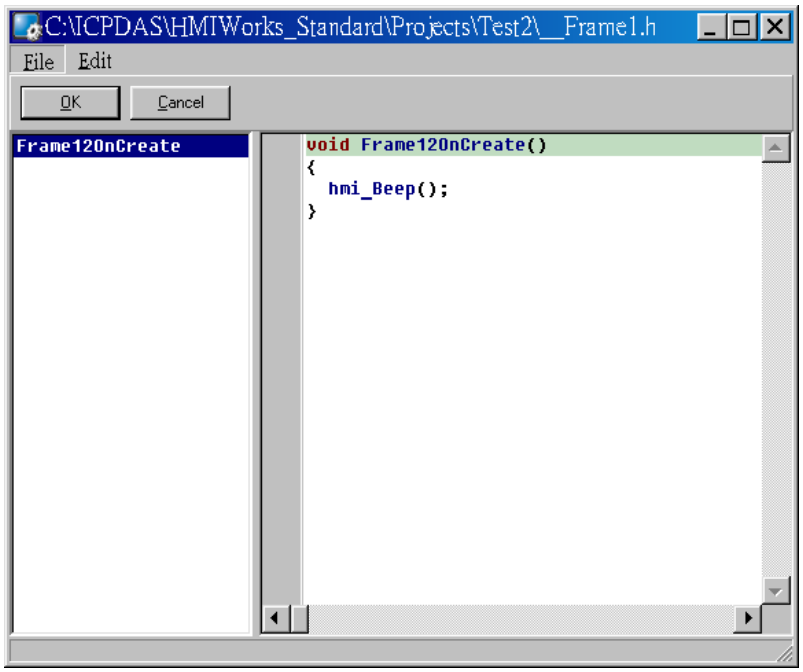
There are 5 event handlers of a frame, OnCreate, OnDestroy, OnHide, OnPaint, and OnShow.

Take OnCreate event handler for example.

1. Click on the OnCreate in the **inspector**. Then click on the button "**…**" to code-edited window for the OnCreate.



2. Here we use hmi_Beep() to sound a beep for example.



3. Save the file, and then press **OK** in the code-edited window.

5.4.2. Drawing a Rectangle

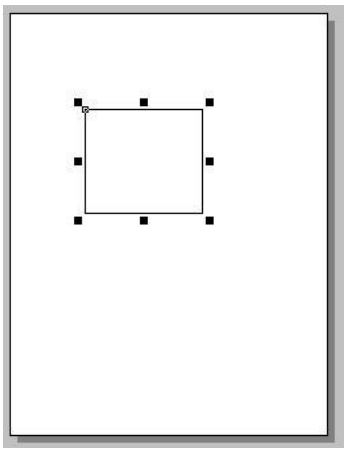
This section shows how to draw a rectangle.

How to draw a rectangle

1. Left-click on the Rectangle icon in the Drawing list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the rectangle to draw.
4. Move the mouse to form a rectangle as you want and stop the

mouse while the cursor is at the point of the right-bottom vertex of the rectangle to draw.

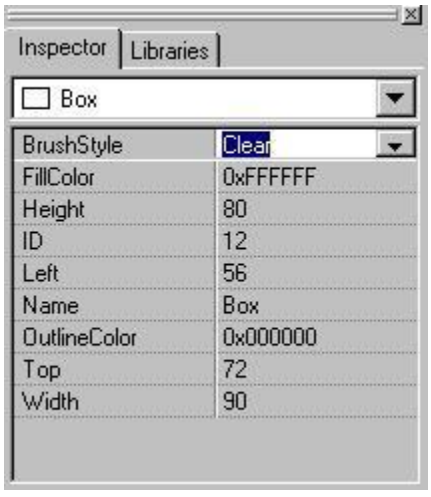
5. Release the left button of the mouse and you have a rectangle you want.



PS. What to do if I want to draw a square?

Step 3 ~ step 5 with the "**Ctrl**" key pressed at the same time.

Properties of Rectangle

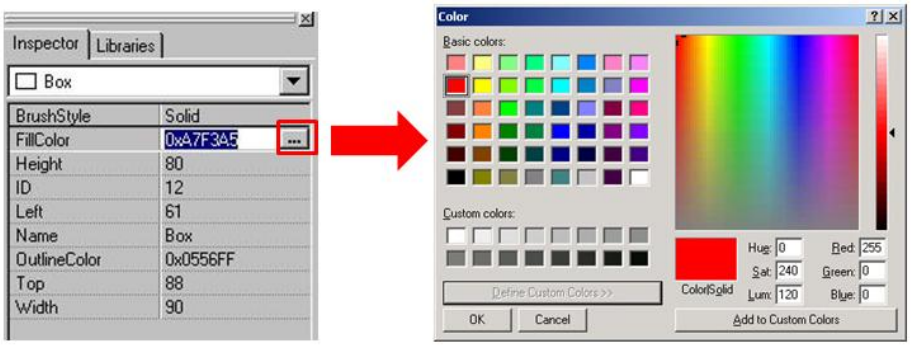


properties	description
BrushStyle	What style to fill to a rectangle
FillColor	What color to fill in the rectangle. The color is represented by a three byte value in hexadecimal form. From the highest byte to

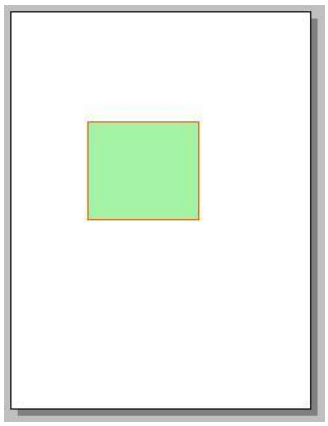
	the lowest, it is the blue byte, the green byte, the red byte in sequence.
Height	Vertical side length of the rectangle
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the rectangle
Name	The name of the rectangle
OutlineColor	The outline color of the rectangle
Top	y-coordinate of the left-top vertex of the rectangle
Width	Horizontal side length of the rectangle

How to change the FillColor and OutlineColor

Click on the FillColor in the **inspector**. Then click on the button "**…**" to open color window to select color. The OutlineColor uses similar way.



For example, set FillColor to green and OutlineColor to red and then you may have the results as shown.

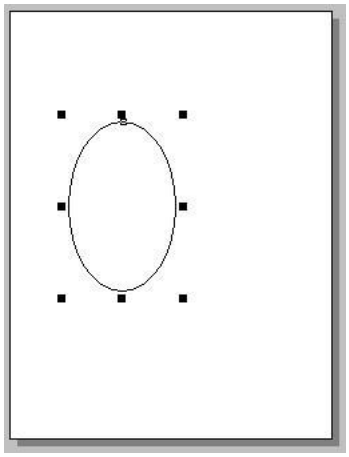


5.4.3. Drawing an Ellipse

This section shows how to draw an ellipse.

How to draw an ellipse

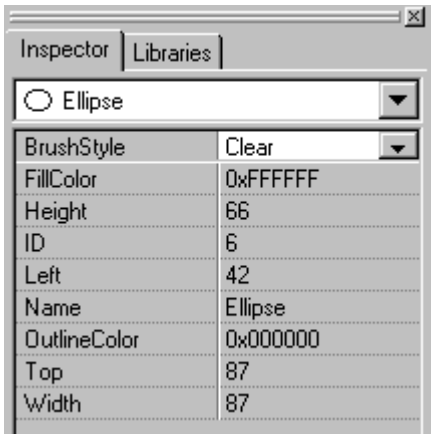
1. Left-click on the Ellipse icon in the Drawing list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the rectangle that inscribes the ellipse to draw.
4. Move the mouse to form the ellipse you want and stop the mouse while the cursor is at the point of the right-bottom vertex of the rectangle that inscribes the ellipse to draw.
5. Release the left button of the mouse and you have an ellipse you want.



PS. What to do if I want to draw a circle?

Step 3 ~ step 5 with the "**Ctrl**" key pressed at the same time.

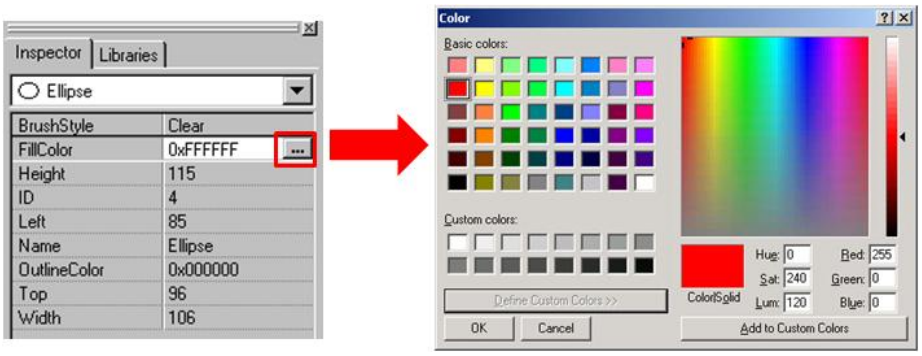
Properties of Ellipse



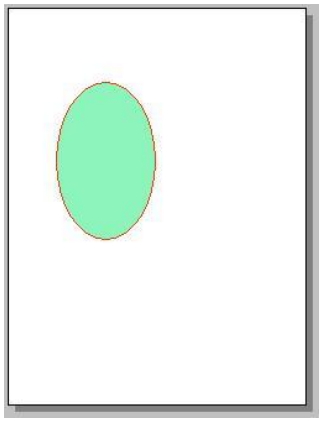
properties	description
BrushStyle	What style to fill to an ellipse
FillColor	What color to fill in the ellipse. The color is represented by a three byte value in hexadecimal form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
Height	Vertical side length of the rectangle that inscribes the ellipse to draw
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the rectangle that inscribes the ellipse to draw
Name	The name of the ellipse
OutlineColor	The outline color of the rectangle that inscribes the ellipse to draw
Top	y-coordinate of the left-top vertex of the rectangle that inscribes the ellipse to draw
Width	Horizontal side length of the rectangle that inscribes the ellipse to draw

How to change the FillColor and OutlineColor

Click on the FillColor in the **inspector**. Then click on the button "**…**" to open color window to select color. The OutlineColor uses similar way.



For example, set FillColor to green and OutlineColor to red and then you may have the results as shown.



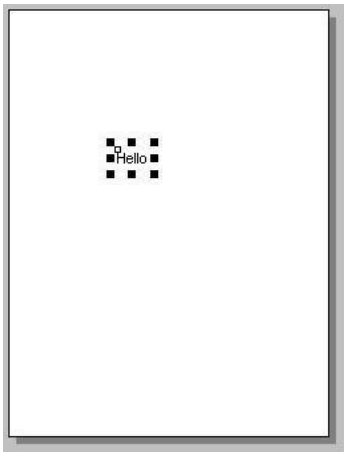
5.4.4. Drawing a Text

This section shows how to draw a Text.

How to draw a Text

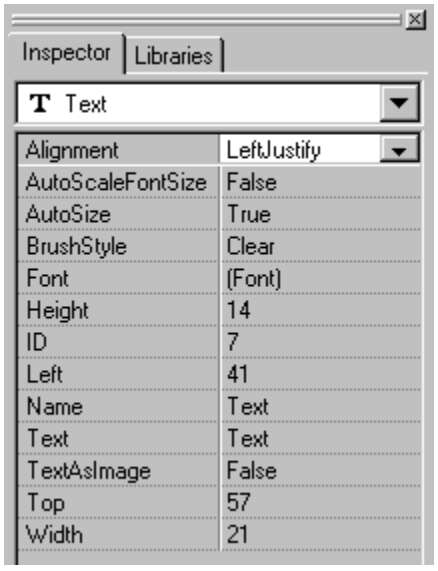
1. Left-click on the Text icon in the Drawing list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click on the point of the frame panel as the left-top vertex of the rectangle that encloses the Text to draw.

4. Change the Text property in the **inspector** to “Hello”, for example.



5. Or you can just copy an text from the clipboard and paste on HMIWorks. HMIWorks create a Text and then load the string from clipboard automatically.

Properties of Text



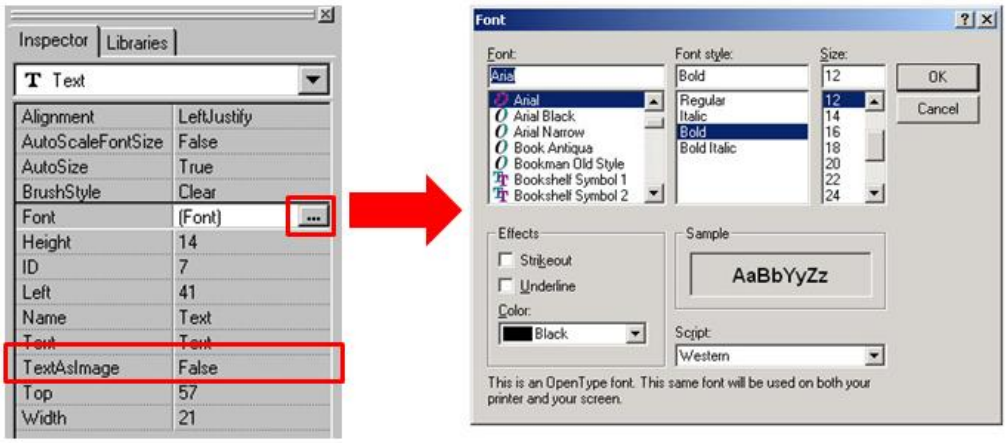
properties	description
Alignment	This property decides which position of the string locates. Left, right, or center. (LeftJustify, RightJustify, or Center) Note: this property is enabled only when AutoSize=True

AutoScaleFontSize	Automatically scale the font size to fit the rectangle which encloses the Text. Note: this property is enabled only when AutoSize=True.
AutoSize	True or False. This property is used to indicate that whether the size of the rectangle which encloses Text can be automatically changed to cover the whole string.
BrushStyle	What style to fill to the rectangle that encloses the Text
Font	The font of the Text. Note: this property is enabled only when TextAsImage=True.
Height	Vertical side length of the rectangle that encloses the Text to draw
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the rectangle that encloses the Text to draw
Name	The name of the Text
Text	The string of the Text to show
TextAsImage	True or False. Whether text is stored as an image or not. If the text is treated as an image, of course, it takes more space to store and more time to download.
Top	y-coordinate of the left-top vertex of the rectangle that encloses the Text to draw
Width	Horizontal side length of the rectangle that encloses the Text to draw

How to change the font of Text

Click on the font property in the **inspector**. Then click on the button "**…**" to open font window to change font.

Be sure to set TextAsImage = True. Otherwise changing font does not take effect.

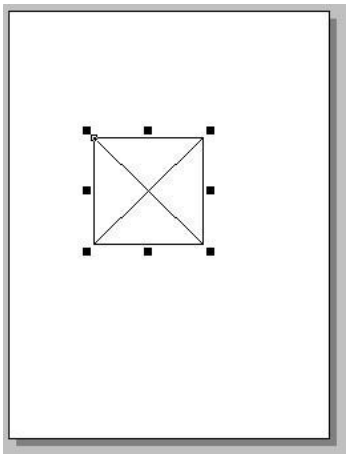


5.4.5. Loading a Picture

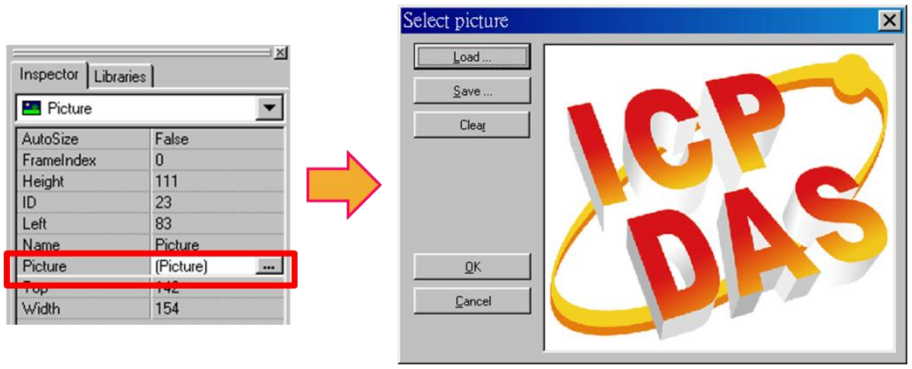
This section shows how to load a Picture.

How to load a Picture

1. Left-click on the Picture icon in the Drawing list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the rectangle that encloses the picture to load.
4. Move the mouse to cover the area where you want to put the picture while the cursor is at the point of the right-bottom vertex of the rectangle that encloses the picture.
5. Release the left button of the mouse and you have an area for the picture.



6. Click on the picture property in the **inspector**. Then click on the button "**…**" to open "**Select picture**" window to load a picture.

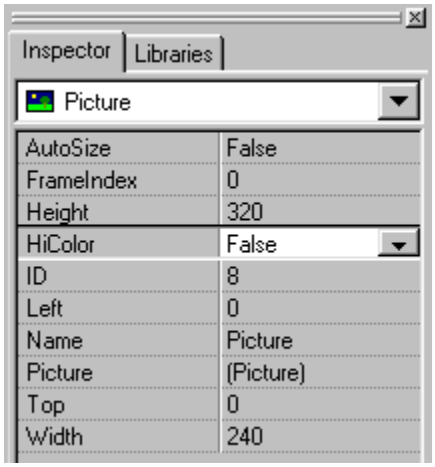


7. The frame with the loaded picture.

8. Or you can just copy an image from the clipboard and paste on

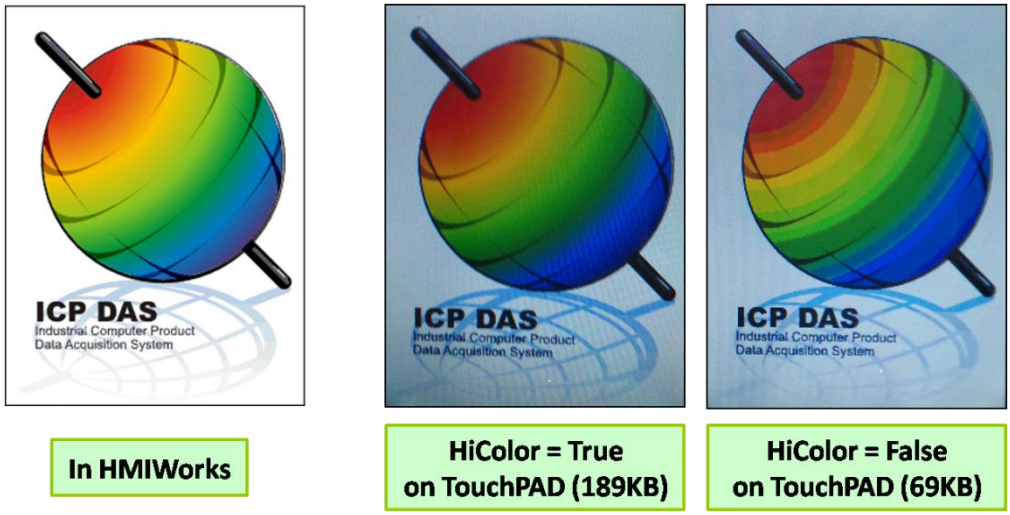
HMIWorks. HMIWorks create a Picture and then load the image from clipboard automatically.

Properties of Picture



properties	description
AutoSize	True or False. This property is used to indicate that whether the size of the Picture can be changed or not.
FrameIndex	Ignored
Height	Vertical side length of the rectangle that encloses the Picture to load
HiColor	True or False. This property decides whether the loaded picture is stored as 16-bit color (True) or 8-bit color (False). Default is 8-bit color.
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the rectangle that encloses the Picture to load
Name	The name of the Picture
Picture	The picture to be loaded
Top	y-coordinate of the left-top vertex of the rectangle that encloses the Picture to load
Width	Horizontal side length of the rectangle that encloses the Picture to load

Trade-off between firmware size and resolution



In HMIWorks	HiColor = True on TouchPAD (189KB)	HiColor = False on TouchPAD (69KB)

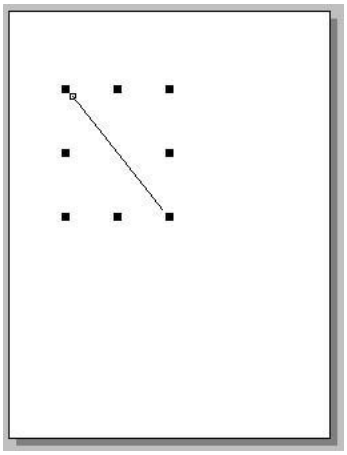
Above is the comparison between HiColor = True and HiColor = False. The left picture is original one in HMIWorks. The two right-side pictures are real photos. One is HiColor = True and the other HiColor = False. As you can see, setting HiColor to False makes the photo have a not-smooth gradient part while setting HiColor to True does not. Because 8-bit color does not have enough color (256 only) to represent the picture, similar colors are represented by the same color and this results in not-smooth gradient.
However, preventing pictures from not-smooth gradient costs TouchPAD bigger size of memory. Take above picture for example, setting HiColor to True uses memory of 189KB but setting HiColor to False costs only 69KB.

5.4.6. Drawing a Line

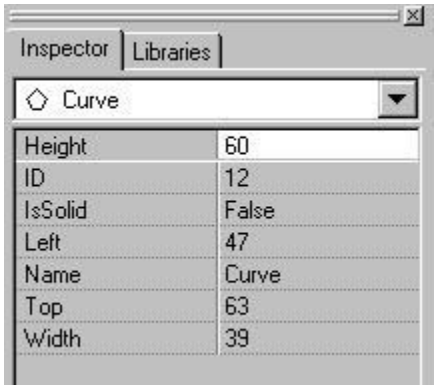
This section shows how to draw a line segment.

How to draw a line segment:

1. Left-click on the Line icon in the Drawing list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the first end point of the line segment to draw.
4. Move the mouse to decide the second end point of the line segment to draw.
5. Release the left button of the mouse and you have a line segment you want.



Properties of Line



properties	description
Height	Vertical side length of the rectangle whose diagonal line is the line segment to draw
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.

IsSolid	Ignored
Left	x-coordinate of the left-top vertex of the rectangle whose diagonal line is the line segment to draw
Name	The name of the line segment
Top	y-coordinate of the left-top vertex of the rectangle whose diagonal line is the line segment to draw
Width	Horizontal side length of the rectangle whose diagonal line is the line segment to draw

5.4.7. Using a TextPushButton

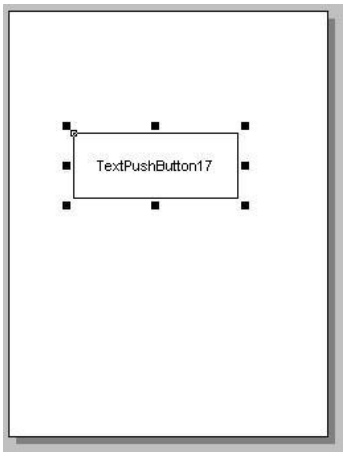
This section shows how to use a TextPushButton.

What is a TextPushButton?

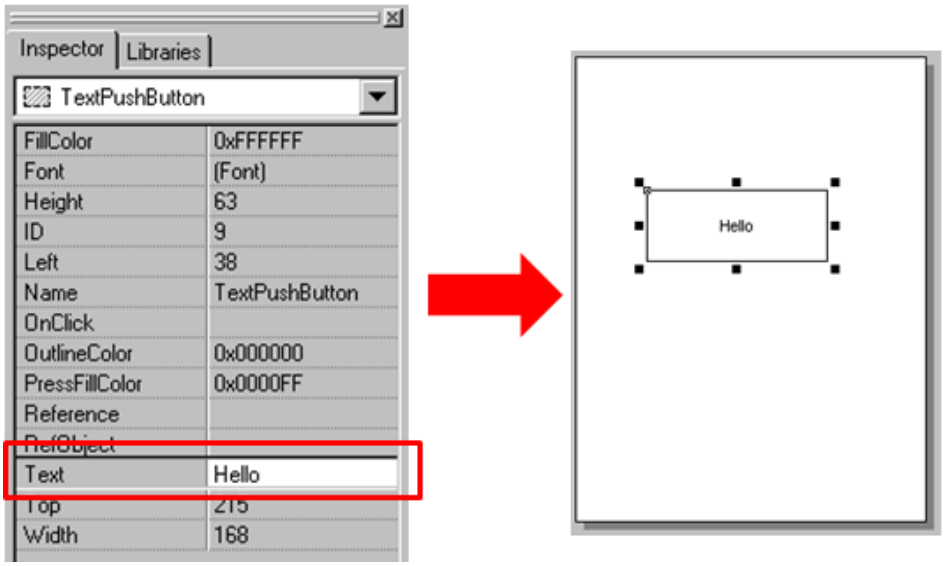
A TextPushButton is a button with a Text on it. When a TextPushButton is pressed and not released, the status is changed. But the status is restored back to the original after you release it.

How to use a TextPushButton:

1. Left-click on the TextPushButton icon in the widget list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the TextPushButton rectangle.
4. Move the mouse to form a TextPushButton as you want and stop the mouse while the cursor is at the point of the right-bottom vertex of the TextPushButton rectangle.
5. Release the left button of the mouse and you have a TextPushButton you want.

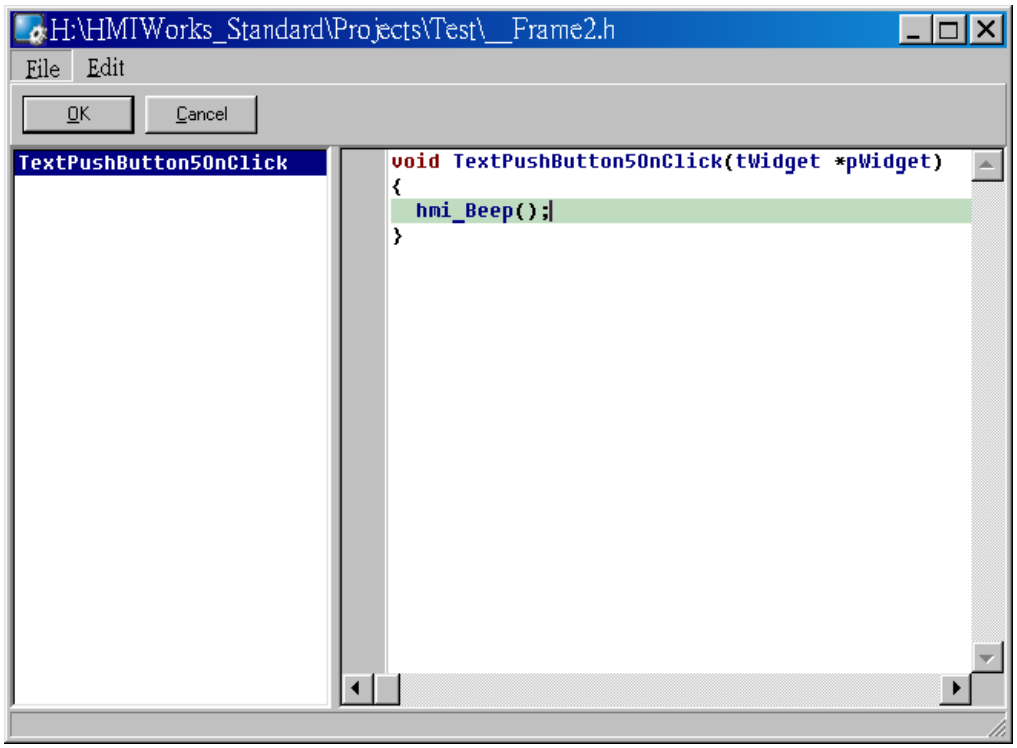


6. Click on the Text property in the **inspector** to change the string on the TextPushButton. Here we change the Text to “Hello”.



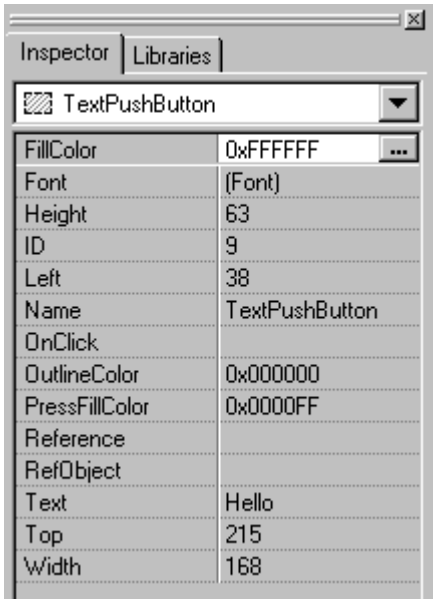
7. Double click on the TextPushButton on the frame panel. Write codes you want in the evoking code-edited window for the On-Click event. Here we use hmi_Beep() to sound a beep for example.

   Note: the property “OnClick” is supported only in programming type “Standard C”.



8. Save the file, and then press **OK** in the code-edited window.

Properties of Text PushButton

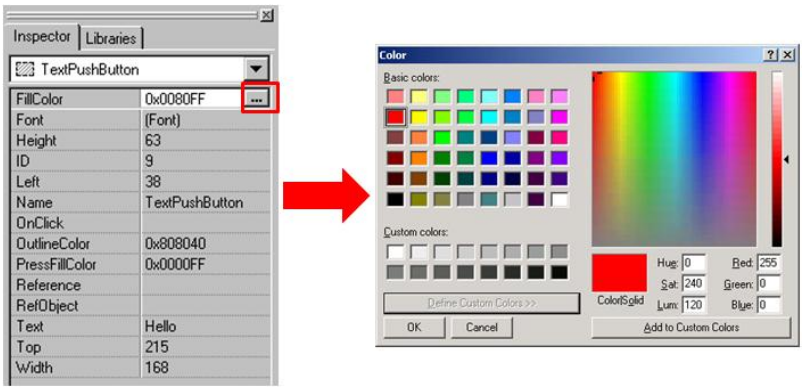


properties	description
FillColor	What color to fill in the TextPushButton. The color

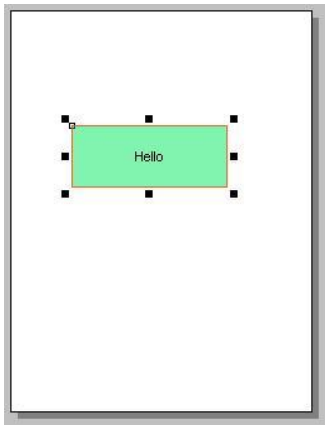
	is represented by a three byte value in hexadecimal form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
Font	The font of the text. Note that the only font style and size are effective.
Height	Vertical side length of the TextPushButton
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the TextPushButton
Name	The name of the TextPushButton
OnClick	The function name of the on-click event of the TextPushButton. **Note**: the property is supported only in programming type “Standard C”.
OutlineColor	The outline color of the TextPushButton
PressFillColor	The color to fill in the TextPushButton when the TextPushButton on the TouchPAD is touched (but not yet released)
Reference	The reference to a frame. That is, when pressing on the TextPushButton, TouchPAD goes to the frame you specified in this property. **Note**: the **priority** of the property “Reference” is higher than that of “OnClick”.
RefObject	The reference to the object list. An object list is a component that can be chosen in the **Toolbox** to maintain a list of the elements of the **library**. Refer to “Using an ObjectList” section for details.
Text	The string of the TextPushButton
Top	y-coordinate of the left-top vertex of the TextPushButton
Width	Horizontal side length of the TextPushButton

How to change FillColor, OutlineColor, and PressFillColor

Click on the FillColor in the **inspector**. Then click on the button "**…**" to open color window to select color. The OutlineColor and PressFillColor use similar way.



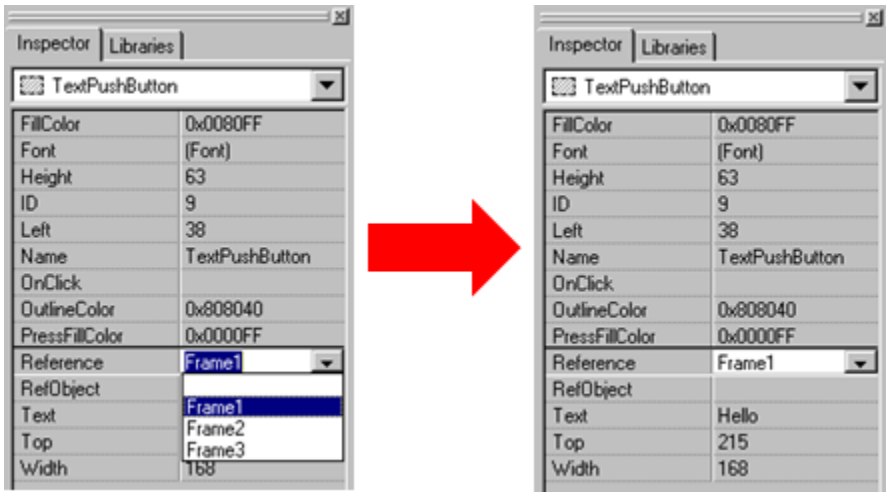
For example, set FillColor to green and OutlineColor to red and then you may have the results as shown.



How to use Reference to another frame

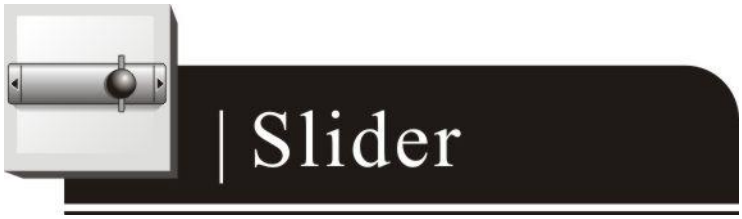
The Reference property is used as a go-to-specified-frame event. **It has higher priority than other events, such as OnClick event.** Thus specifying a value to the Reference property disables the OnClick event.

It's easy to specify a value to the Reference property. Simply click on the Reference property in the **inspector** and then choose the frame for reference.



5.4.8. Using a Slider

This section shows how to use a Slider.



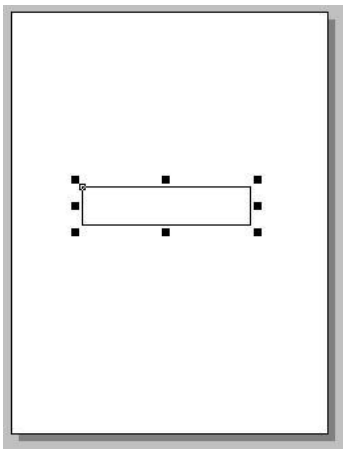
What is a Slider?

A Slider is a control element used to set levels. Usually, a Slider is used in volume control.

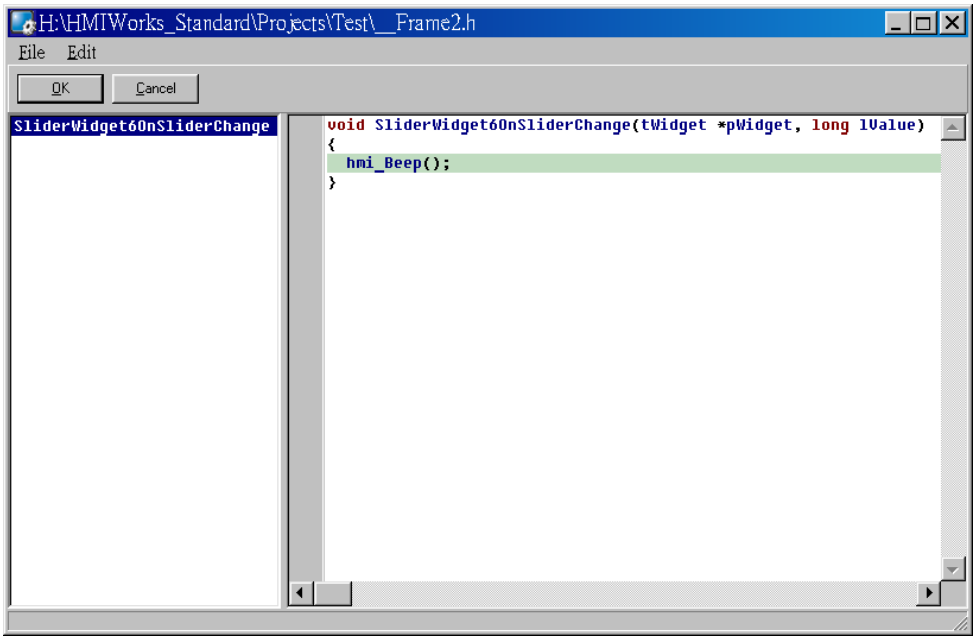
How to use a Slider:

1. Left-click on the Slider icon in the widget list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the Slider rectangle.
4. Move the mouse to form a Slider as you want and stop the mouse while the cursor is at the point of the right-bottom vertex of the Slider rectangle.
5. Release the left button of the mouse and you have a Slider you

want.

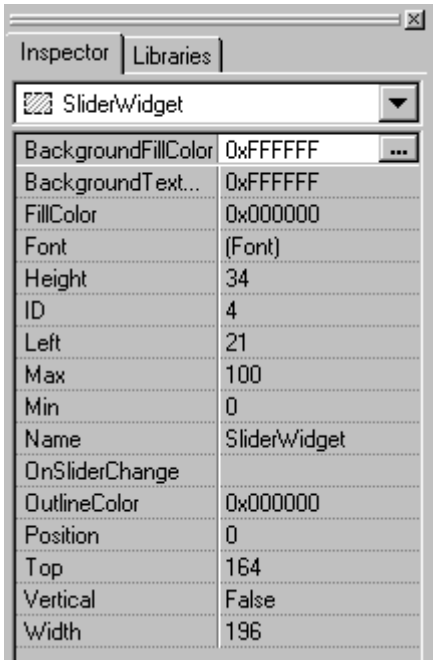


6. Double click on the Slider on the frame panel. Write codes you want in the evoking code-edited window for the OnSliderChange event. Here we use hmi_Beep() to sound a beep for example.
   Note: the property “OnSliderChange” is supported only in programming type “Standard C”.



7. Save the file, and then press **OK** in the code-edited window.

Properties of Slider

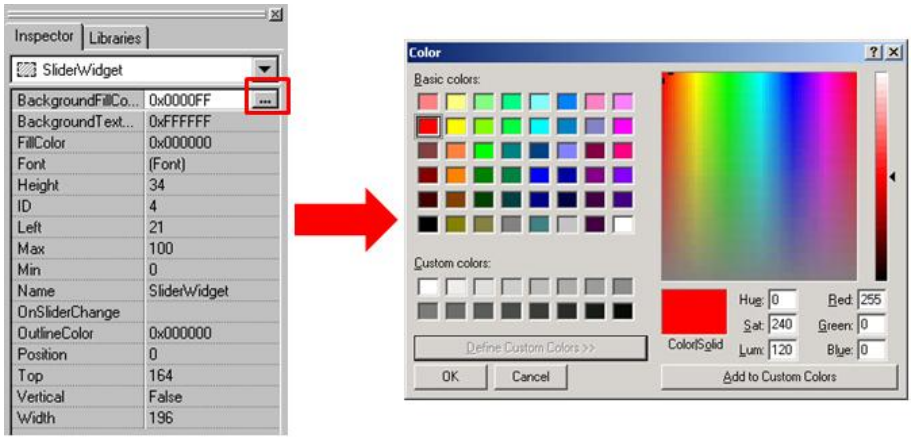


properties	description
BackgroundFillColor	What color to fill in the background of the Slider. The color is represented by a three byte value in hexadecimal form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
BackgroundTextColor	What's color of the text in the background of the Slider. The color is represented by a three byte value in hexadecimal form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
FillColor	What color to fill in the Slider. The color is represented by a three byte value in hexadecimal form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
Font	The font of the text on the Slider

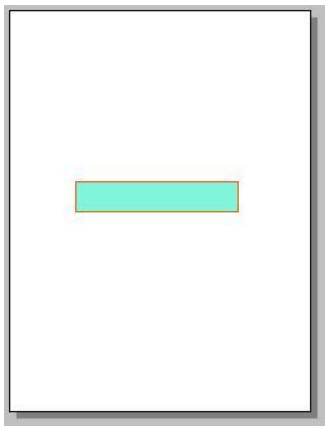
Height	Vertical side length of the Slider
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the Slider
Max	The maximum value of the Position
Min	The minimum value of the Position
Name	The name of the Slider
OnSliderChange	The function name of the on-slider-change event of the Slider. **Note**: the property is supported only in programming type “Standard C”.
OutlineColor	The outline color of the Slider
Position	The value where the slider locate (between Max and Min)
TagName	Associate a variable (tag) in **Ladder Designer**. **Note**: the property is supported only in programming type “Ladder”.
Top	y-coordinate of the left-top vertex of the Slider
Vertical	The direction of the Slider
Width	Horizontal side length of the Slider

How to change FillColor and OutlineColor

Click on the FillColor in the **inspector**. Then click on the button “**…**” to open color window to select color. The OutlineColor, BackgroundFillColor and BackgroundTextColor use similar way.



For example, set FillColor to green and OutlineColor to red and then you may have the results as shown.



5.4.9. Using a BitButton

This section shows how to use a BitButton.

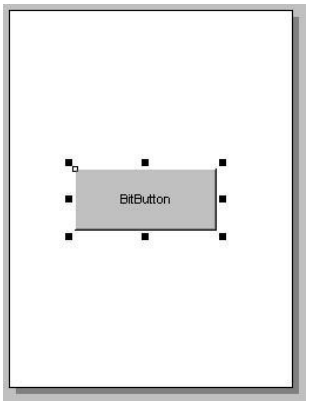
What is a BitButton?

A BitButton is a button with 3D appearance and the status rebounds back if releasing the pressed button. When you press it, you can see

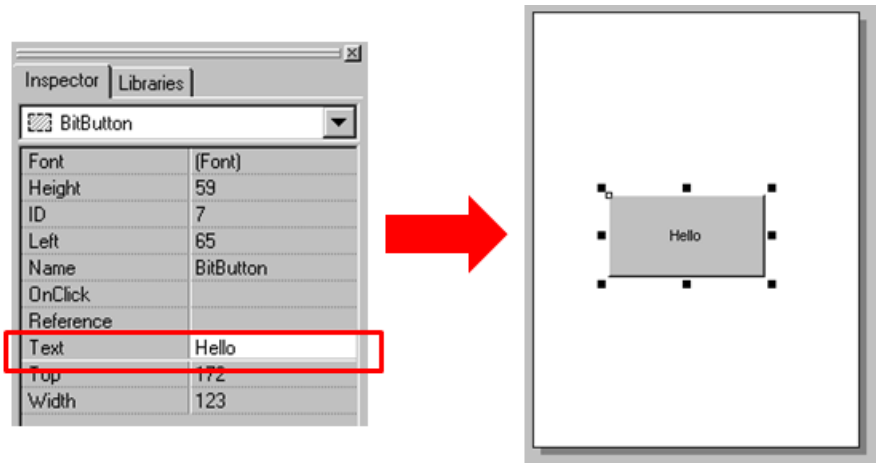
that the BitButton is pressed “down”. This 3D-like appearance is achieved by two images so that it takes more spaces to store and more time to download than a Text PushButton.

How to use a BitButton:

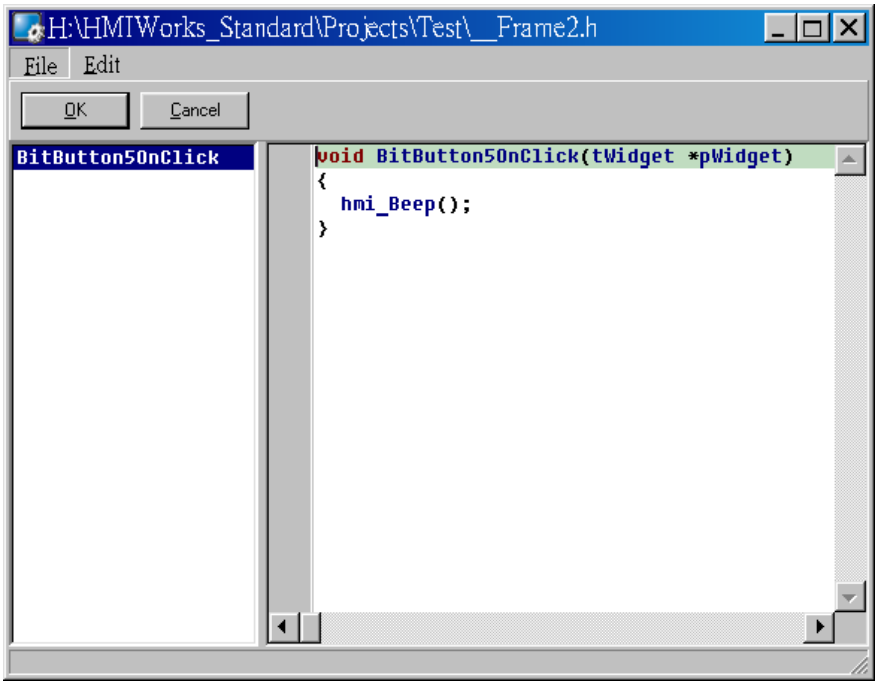
1. Left-click on the BitButton icon in the widget list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the BitButton.
4. Move the mouse to form a BitButton as you want and stop the mouse while the cursor is at the point of the right-bottom vertex of the BitButton.
5. Release the left button of the mouse and you have a BitButton you want.



6. Click on the Text property in the **inspector** to change the string on the BitButton. Here we change the Text to “Hello”.

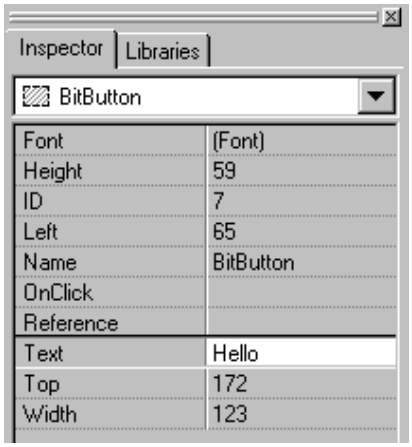


7. Double click on the BitButton on the frame panel. Write codes you want in the evoking code-edited window for the On-Click event. Here we use hmi_Beep() to sound a beep for example. **Note**: the property “OnClick” is supported only in programming type “Standard C”.



8. Save the file, and then press **OK** in the code-edited window.

Properties of BitButton



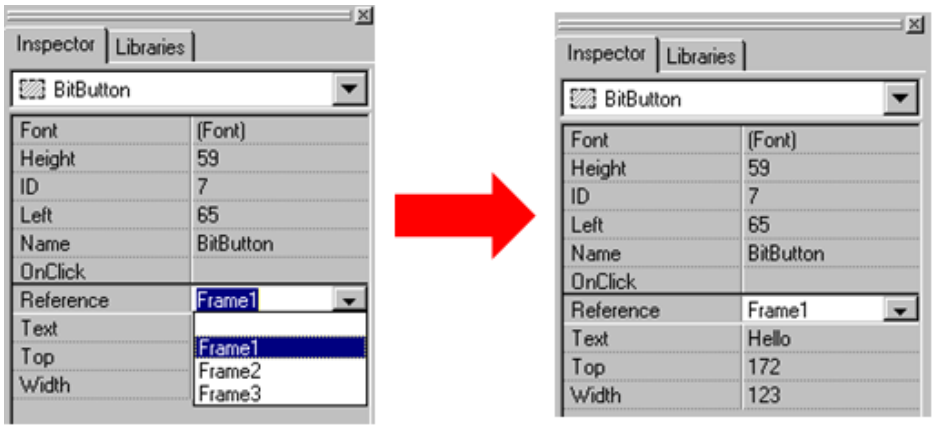
properties	description
Font	The font of the text of the BitButton

Height	Vertical side length of the BitButton
ID	The serial number of tools
Left	x-coordinate of the left-top vertex of the BitButton
Name	The name of the BitButton
OnClick	The function name of the on-click event of the BitButton. **Note**: the property is supported only in programming type “Standard C”.
Reference	The reference to a frame. That is, when pressing on the BitButton, TouchPAD goes to the frame you specified in this property. **Note**: the **priority** of the property “Reference” is higher than that of “OnClick”.
Text	The string on the BitButton
Top	y-coordinate of the left-top vertex of the BitButton
Width	Horizontal side length of the BitButton

How to use Reference to another frame

The Reference property is used as a go-to-specified-frame event.
It has higher priority than other events, such as OnClick event.
Thus specifying a value to the Reference property disables the OnClick event.

It’s easy to specify a value to the Reference property. Simply click on the Reference property in the **inspector** and then choose the frame for reference.



5.4.10. Using a HotSpot

This section shows how to use a HotSpot.

What is a HotSpot?

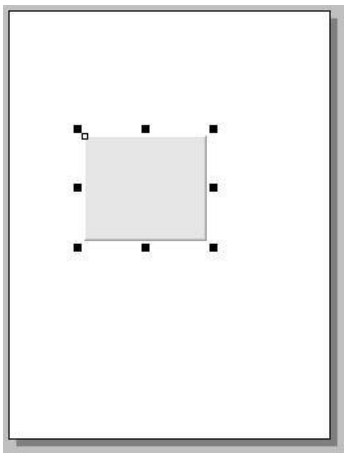
HotSpot decides an area which is capable of responding to on-click events. Usually, putting a HotSpot on the Drawing components (that is, Rectangle, Ellipse, Text, Picture, and Line) makes them to respond to on-click events. After downloading to TouchPAD, a HotSpot is invisible.

How to use a HotSpot:

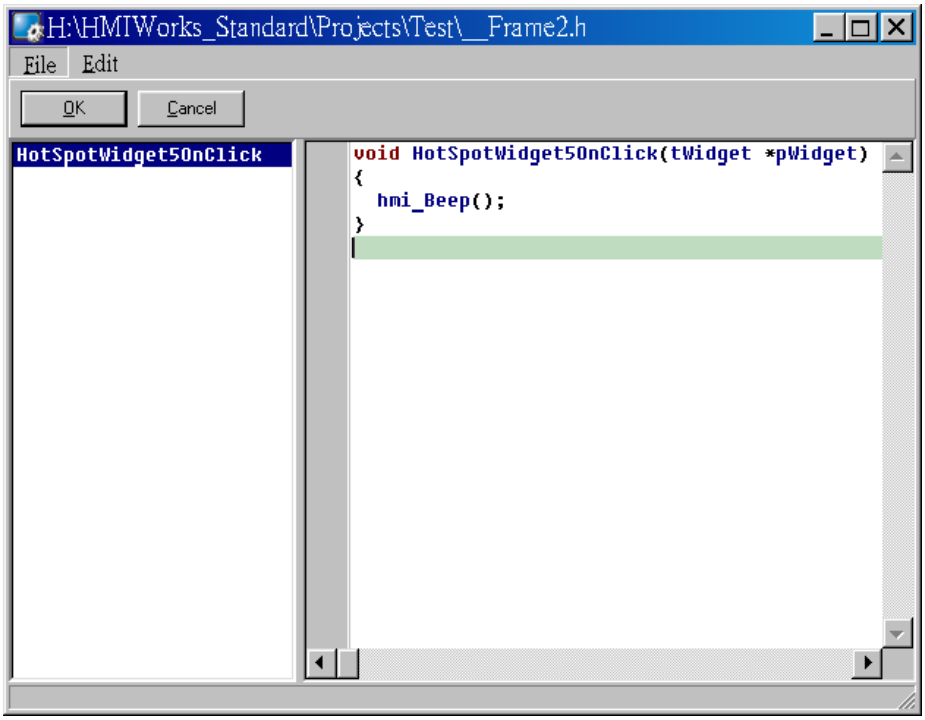
1. Left-click on the HotSpot icon in the widget list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the HotSpot rectangle.
4. Move the mouse to form a HotSpot as you want and stop the mouse while the cursor is at the point of the right-bottom vertex

of the HotSpot rectangle.

5. Release the left button of the mouse and you have a HotSpot you want.

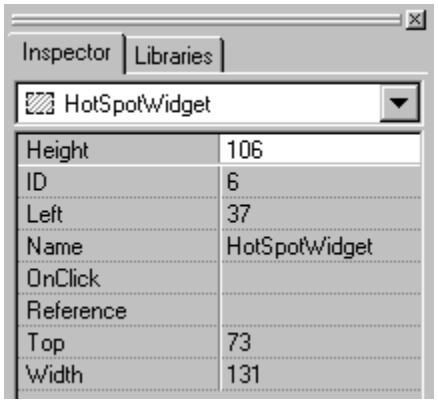


6. Double click on the HotSpot on the frame panel. Write codes you want in the evoking code-edited window for the On-Click event. Here we use hmi_Beep() to sound a beep for example. **Note**: the property “OnClick” is supported only in programming type “Standard C”.



7. Save the file, and then press **OK** in the code-edited window.

Properties of HotSpot



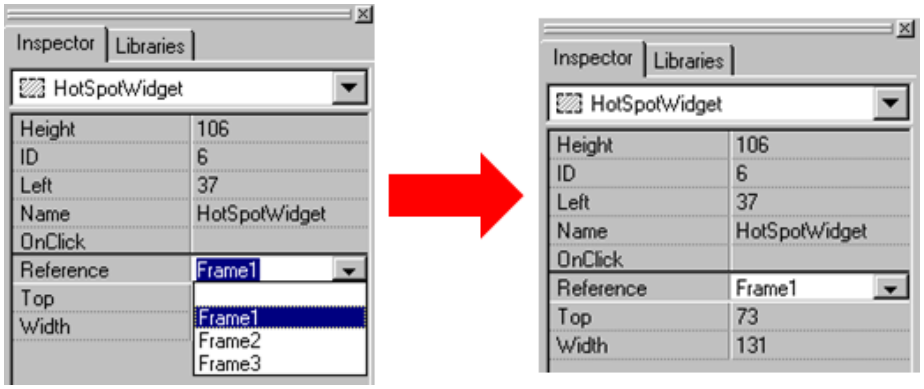
properties	description
Height	Vertical side length of the HotSpot
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the HotSpot
Name	The name of the HotSpot
OnClick	The function name of the on-click event of the HotSpot. **Note**: the property is supported only in programming type “Standard C”.
Reference	The reference to a frame. That is, when pressing on the HotSpot, TouchPAD goes to the frame you specified in this property. **Note**: the **priority** of the property “Reference” is higher than that of “OnClick”.
Top	y-coordinate of the left-top vertex of the HotSpot
Width	Horizontal side length of the HotSpot

How to use Reference to another frame

The Reference property is used as a go-to-specified-frame event.
It has higher priority than other events, such as OnClick event.
Thus specifying a value to the Reference property disables the OnClick event.

It’s easy to specify a value to the Reference property. Simply click on the Reference property in the **inspector** and then choose the frame for

reference.



5.4.11. Using a CheckBox

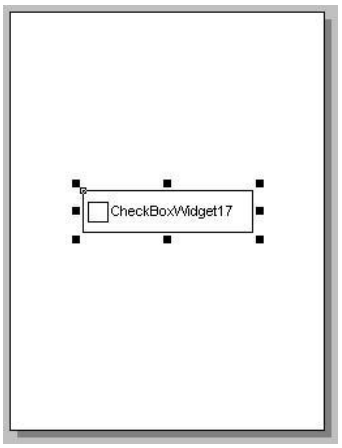
This section shows how to use a CheckBox.

What is a CheckBox?

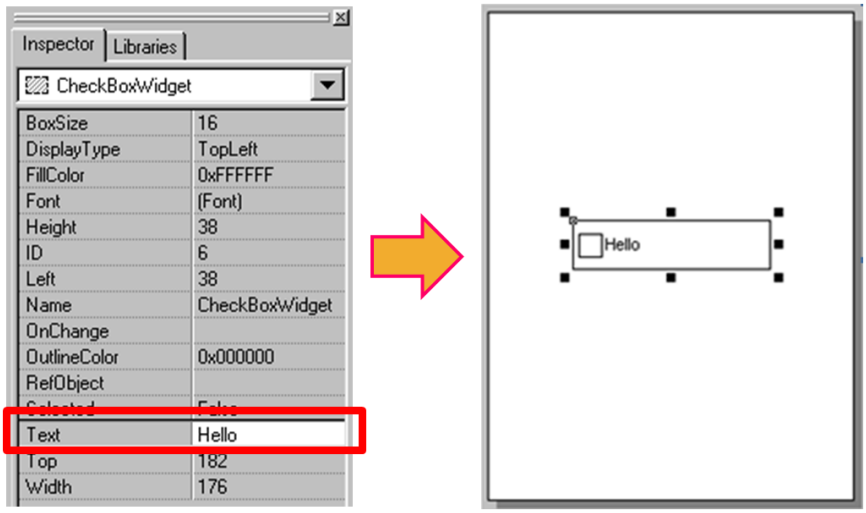
A CheckBox is a control element that provides a yes-no choice.

How to use a CheckBox:

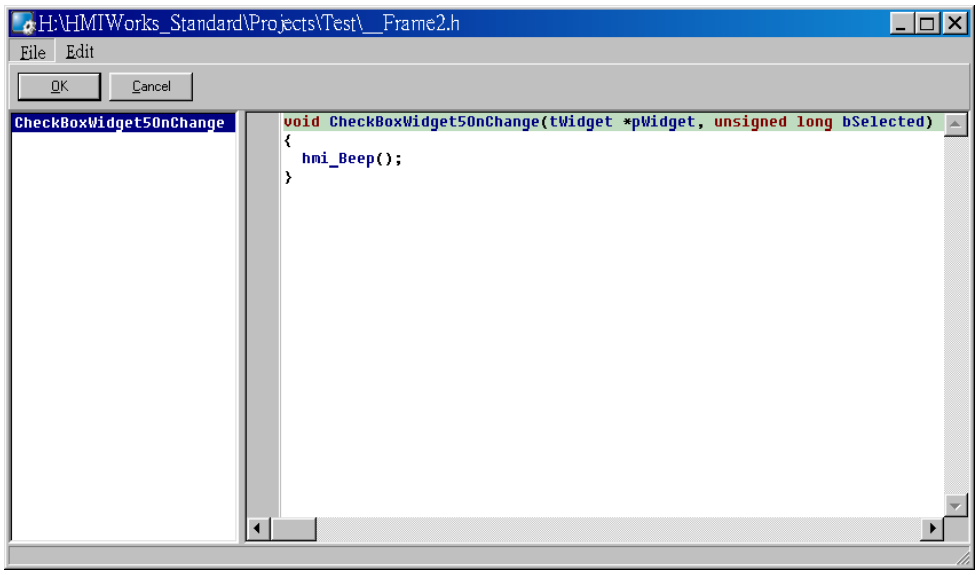
1. Left-click on the CheckBox icon in the widget list of the Toolbox.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the CheckBox rectangle.
4. Move the mouse to form a CheckBox as you want and stop the mouse while the cursor is at the point of the right-bottom vertex of the CheckBox rectangle.
5. Release the left button of the mouse and you have a CheckBox you want.



6. Click on the Text property in the **inspector** to change the string on the CheckBox. Here we change the Text to “Hello”.

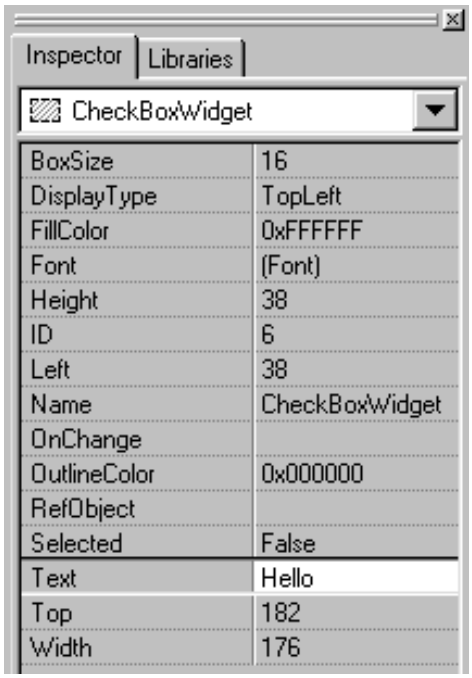


7. Double click on the CheckBox on the frame panel. Write codes you want in the evoking code-edited window for the OnChange event. Here we use hmi_Beep() to sound a beep for example.
Note: the property “OnChange” is supported only in programming type “Standard C”.



8. Save the file, and then press **OK** in the code-edited window.

Properties of CheckBox

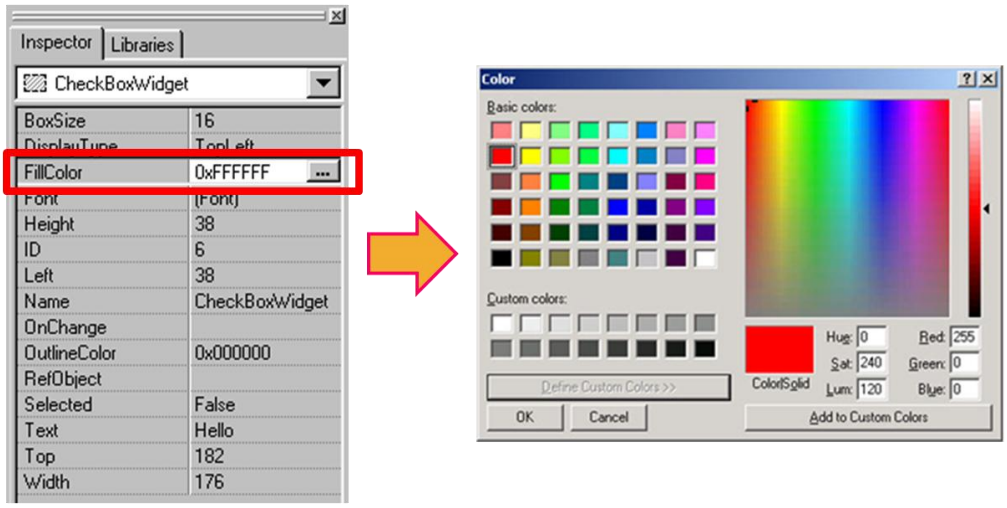


properties	description
BoxSize	The size of the checking box
DisplayType	How to display the pictures which are loaded from RefObject property.
FillColor	What color to fill in the CheckBox. The color is represented by a three byte value in hexadecimal

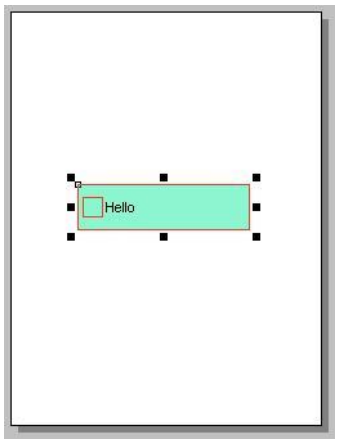
	form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
Font	The font of the text of the CheckBox
Height	Vertical side length of the CheckBox
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the CheckBox
Name	The name of the CheckBox
OnChange	The function name of the OnChange event of the CheckBox.
OutlineColor	The outline color of the CheckBox
RefObject	The reference to the object list. An object list is a component that can be chosen in the **Toolbox** to maintain a list of the elements of the library. Refer to section “Using an ObjectList” for details.
Selected	True or false. Whether the CheckBox is checked or not
TagName	Associate a variable (tag) in **Ladder Designer**. **Note**: the property is supported only in programming type “Ladder”.
Text	The string of the CheckBox
Top	y-coordinate of the left-top vertex of the CheckBox
Width	Horizontal side length of the CheckBox

How to change FillColor and OutlineColor

Click on the FillColor in the **inspector**. Then click on the button “**…**” to open color window to select color. The OutlineColor uses similar way.



For example, set FillColor to green and OutlineColor to red and then you may have the results as shown.



5.4.12. Using a Label

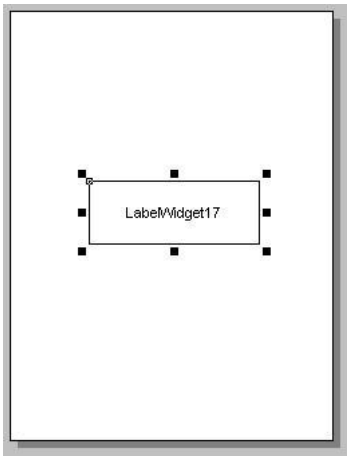
This section shows how to use a Label.

What is a Label?

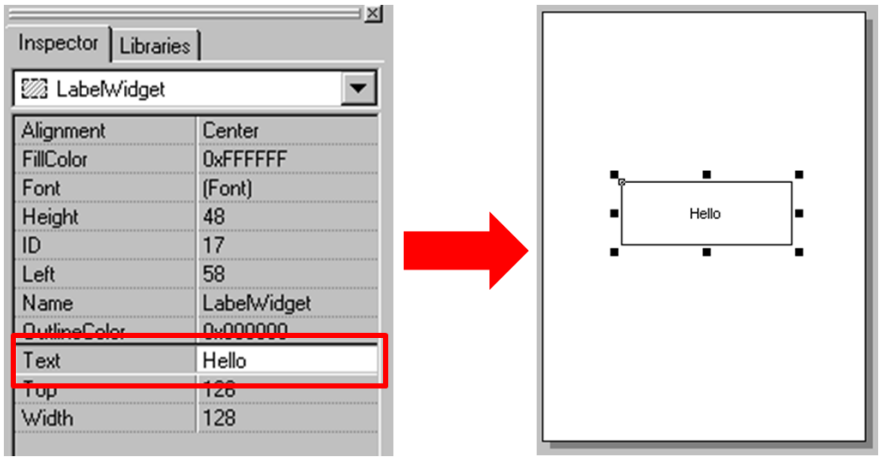
A Label is a Text put on TouchPAD to give information that may change at run time.

How to use a Label:

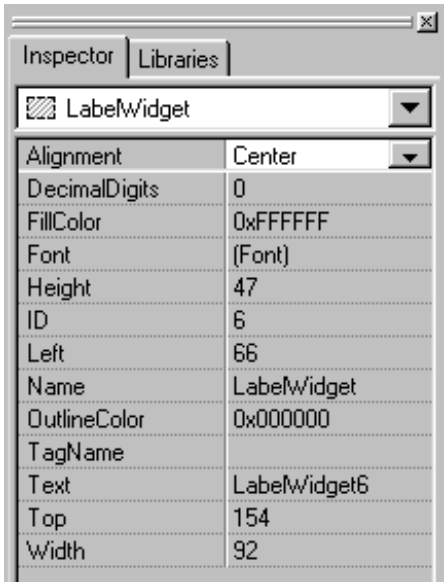
1. Left-click on the Label icon in the widget list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the Label rectangle.
4. Move the mouse to form a Label as you want and stop the mouse while the cursor is at the point of the right-bottom vertex of the Label rectangle.
5. Release the left button of the mouse and you have a Label you want.



6. Click on the Text property in the **inspector** to change the string on the Label. Here we change the Text to “Hello”.



Properties of Label

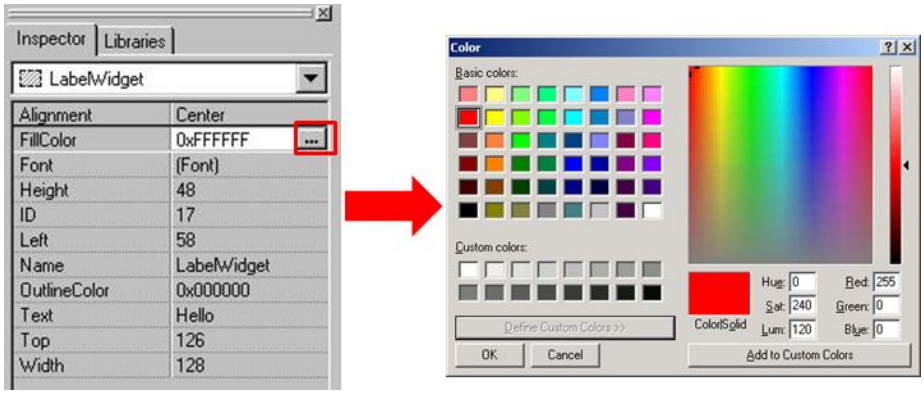


properties	description
Alignment	This property decides which position of the string locates. Left, right, or center. (LeftJustify, RightJustify, or Center)
DecimalDigits	The power to which ten must be raised to produce the value, say divisor, which is used to divide the value of the associated tag of this Label. The value of the tag must be divided by the divisor to show on the screen to represent decimal digits. **Note**: the property is supported only in programming type “Ladder”.
FillColor	What color to fill in the Label. The color is represented by a three byte value in hexadecimal form. From the highest byte to the lowest, it is the blue byte, the green byte, the red byte in sequence.
Font	The font of the text
Height	Vertical side length of the Label
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the Label
Name	The name of the Label
OutlineColor	The outline color of the Label
TagName	Associate a variable (tag) in **Ladder Designer**. **Note**: the property is supported only in programming type “Ladder”.

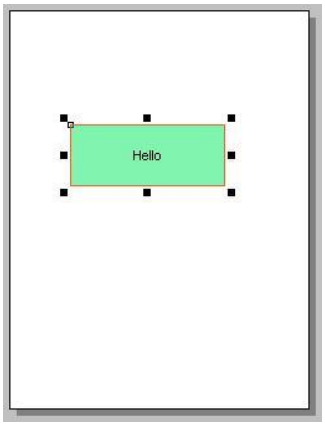
Text	The string of the Label
Top	y-coordinate of the left-top vertex of the Label
Width	Horizontal side length of the Label

How to change FillColor and OutlineColor

Click on the FillColor in the **inspector**. Then click on the button "**…**" to open color window to select color. The OutlineColor uses similar way.



For example, set FillColor to green and OutlineColor to red and then you may have the results as shown.



How to represent decimals for Ladder Designer

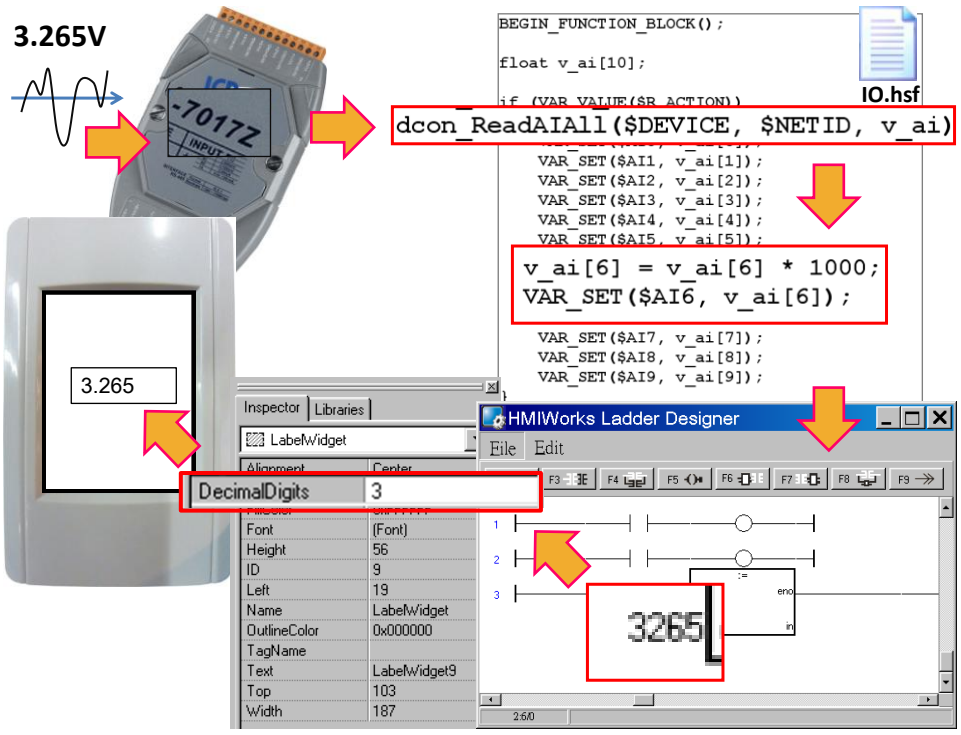
The numbers in **Ladder Desinger** are all integers. The decimals are not accepted in **Ladder Designer**. However, in some cases, users may need to calculate or display decimals. Here we provide a work-around method to handle decimals.

Take I-7017Z for example. Supposed that we use I-7017Z to read an

analog value 3.265V back from a remote equipment and we want to utilize the powerful feature of HMIWorks, **Ladder Designer**. However, **Ladder Designer** supports only integers. So we must handle this drawback to directly read back the AI value from I-7017Z in **Ladder Designer**.

1. Set the property “DecimalDigits” to the number of digits in the right of the decimal point. For example, we set DecimalDigits = 3.
2. Modify the I/O module’s IO.hsf. Let the read back AI value multiplied by ten of the n-th power where n is the value of “DecimalDigits”. You can find out I/O module’s IO.hsf file in the following locations: “[HMIWorks_install_path]\ bin\Modules\”. For example, IO.hsf of I-7017Z is located in “C:\ICPDAS\HMIWorks_Standard\bin\Modules\I-7000\I-7017Z”, where “C:\ICPDAS\HMIWorks_Standard\” is the installation path of HMIWorks. And we modify the IO.hsf to make v_ai[6] = v_ai[6] * 1000; Supposed we use channel 6 to read back AI value.

As the figure shows, you can see that the tag “$AI6” in **Ladder Designer** is 1000 times of the real value. With DecimalDigits set to 3, the correct value 3.265 is displayed on TouchPAD.



5.4.13. Using a Timer

This section shows how to use a Timer.

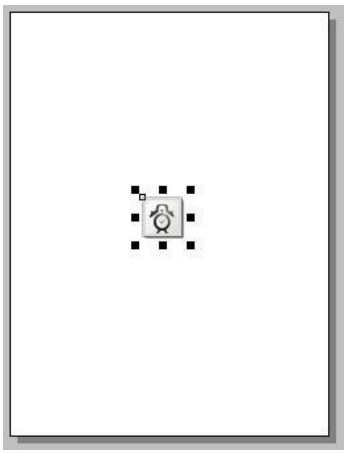
Note: this tool is supported only in programming type “Standard C”.

What is a Timer?

A Timer is a component that executes the OnExecute event handler every specified interval.

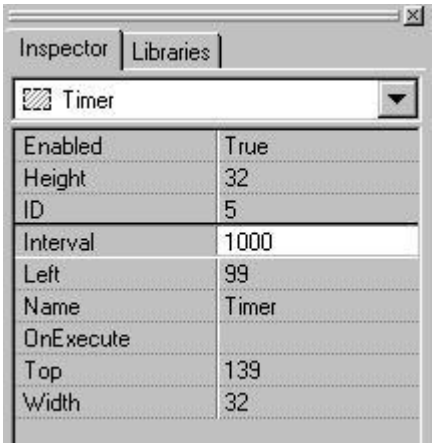
How to use a Timer:

1. Left-click on the Timer icon in the system list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click on the frame panel. Note that you should not worry about the size or the location of the Timer because the Timer is invisible when downloaded to the TouchPAD. Also it’s not necessary to put the Timer on the frame panel.

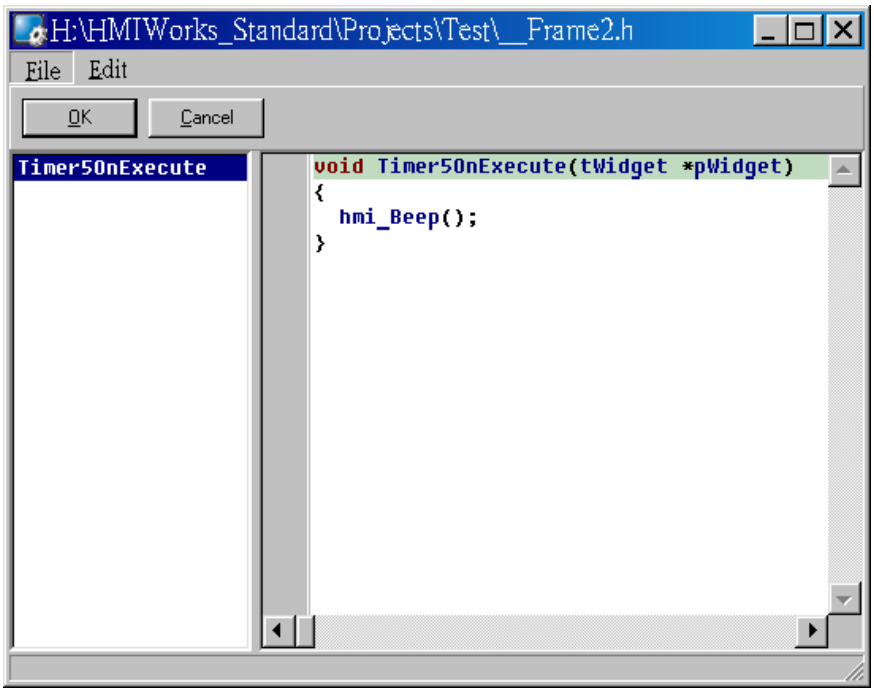


4. Click on the Interval property in the **inspector** to change the

repeating period of the Timer. Here we set the Interval to 1000 (ms). And then change the property “Enabled” to True.

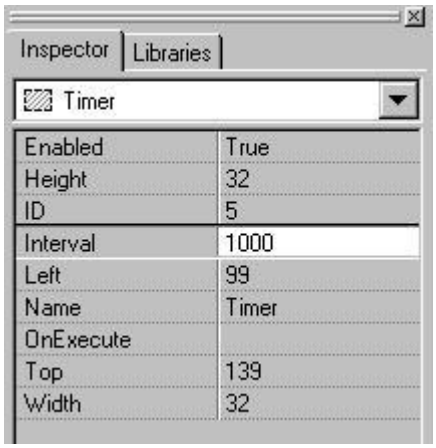


5. Double click on the Timer on the frame panel. Write codes you want in the evoking code-edited window for the OnExecute event. Here we use hmi_Beep() to sound a beep for example.



6. Save the file, and then press **OK** in the code-edited window.

Properties of Timer



properties	description
Enabled	Whether the Timer is enabled or not
Height	Vertical side length of the Timer (This property cannot be changed by users.)
ID	The serial number of tools (including Drawings, Widgets, System component) which is used to identify them.
Interval	The time span of two consecutive OnExecute events
Left	x-coordinate of the left-top vertex of the Timer
Name	The name of the Timer
OnExecute	The function name of the OnExecute event of the Timer.
Top	y-coordinate of the left-top vertex of the Timer
Width	Horizontal side length of the Timer (cannot be changed by users.)

5.4.14. Using a PaintBox

This section shows how to use a PaintBox.

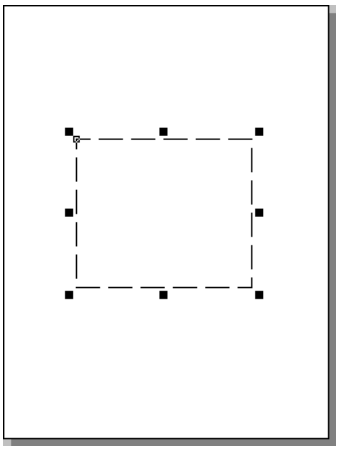
Note: this tool is supported only in programming type “Standard C”.

What is a PaintBox?

A PaintBox is a component which is used to paint shapes, such as rectangles, ellipses, etc, in the runtime.

How to use a PaintBox:

1. Left-click on the PaintBox icon in the system list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click (and not release) on the point of the frame panel as the left-top vertex of the PaintBox rectangle.
4. Move the mouse to form a PaintBox as you want and stop the mouse while the cursor is at the point of the right-bottom vertex of the PaintBox.
5. Release the left button of the mouse and you have a PaintBox you want.

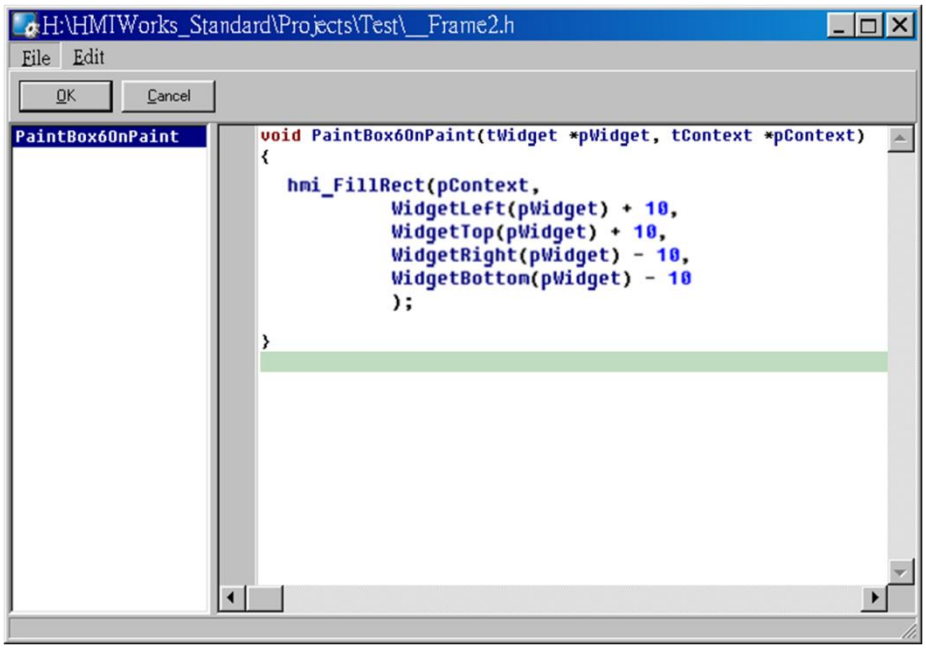


6. Double click on the PaintBox on the frame panel. Write codes you want in the evoking code-edited window for the OnPaint event. Here we draw a rectangle for example.
   Note 1: the diagonal points used in the function of hmi_FillRect

are in **the same** coordinate as the frame.

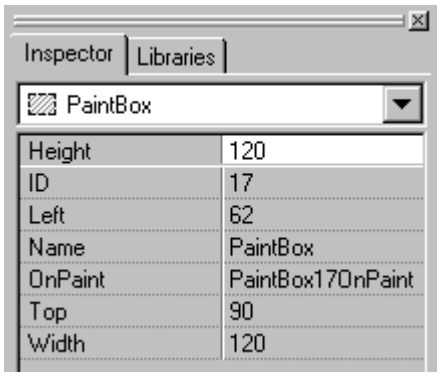
Note 2: the part of the rectangle which is outside the perimeter of the PaintBox is cut off.

Note 3: WidgetLeft(pWidget) and WidgetTop(pWidget) are the x, y coordinates of the left-top vertex of the PaintBox. While WidgetRight(pWidget) and WidgetBottom(pWidget) are those of the right-bottom.



7. Save the file, and then press **OK** in the code-edited window.

Properties of PaintBox



properties	description
Height	Vertical side length of the PaintBox
ID	The serial number of tools (including Drawings and

	Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the PaintBox
Name	The name of the PaintBox
OnPaint	The function name of the OnPaint event of the PaintBox.
Top	y-coordinate of the left-top vertex of the PaintBox
Width	Horizontal side length of the PaintBox

5.4.15. Using an ObjectList

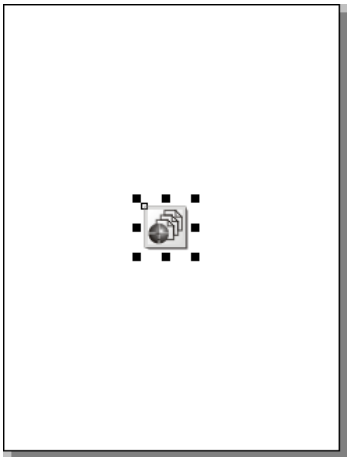
This section shows how to use an ObjectList.

What is an ObjectList?

An ObjectList is a component which is used to maintain a list of library objects (usually two library objects). An ObjectList can be used in both programming type. Combined with “RefObject” properties of the widget TextPushButton and CheckBox, users can easily toggle two images.
Note: TextPushButton does not have the property, TagName, so it is not used in programming type Ladder.

How to use an ObjectList:

1. Left-click on the ObjectList icon in the System list of the **Toolbox**.
2. Move the cursor over the frame panel.
3. Left-click on the frame panel. Note that you should not worry about the size or the location of the ObjectList because the ObjectList is invisible when downloaded to the TouchPAD. Also it’s not necessary to put the ObjectList on the frame panel.

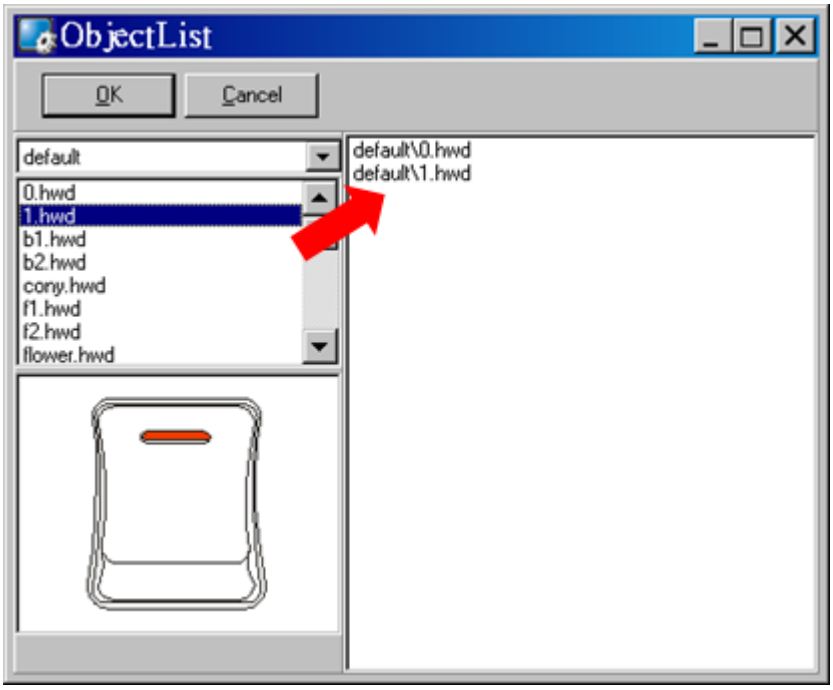


4. The ObjectList maintains a list of a library objects and is used in a TextPushButton or a CheckBox. After downloading to the TouchPAD, the images of the library objects replace the TextPushButton or the CheckBox. When the state of the TextPushButton or the CheckBox changed, users see only the images of the library objects toggles but do not see the original appearances of state exchanges of the TextPushButton or the CheckBox.

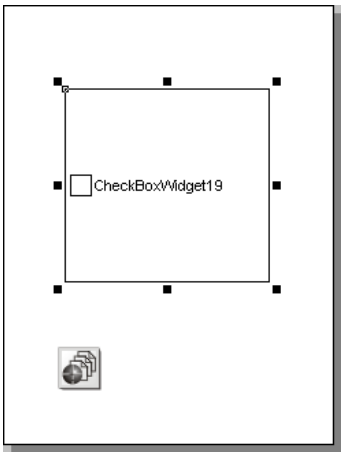
5. Add two library objects in the ObjectList by double-click on the ObjectList icon. Then the window of ObjectList shows. Double click on the list of the library objects adds the clicked library object to the right side region.

   Note 1: you only need to add two library objects to the right side region, because both TextPushButton and CheckBox have only two states.

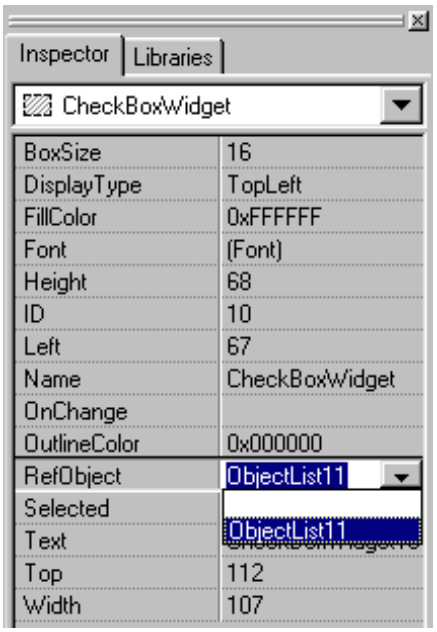
   Note 2: to delete the library objects in the ObjectList window, double click on the objects in the right-side block.



6. Place a CheckBox on the frame for example. Be sure to make the size of the CheckBox large enough to cover the whole image of the library object.

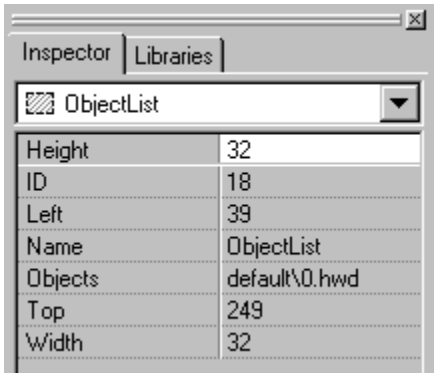


7. Go to the **inspector** for the RefObject property of the CheckBox. Choose the ObjectList to make connections between the ObjectList and the CheckBox.



8. Build and download the project. You can see two images of the library objects toggle and no more the TextPushButton and CheckBox.

Properties of ObjectList

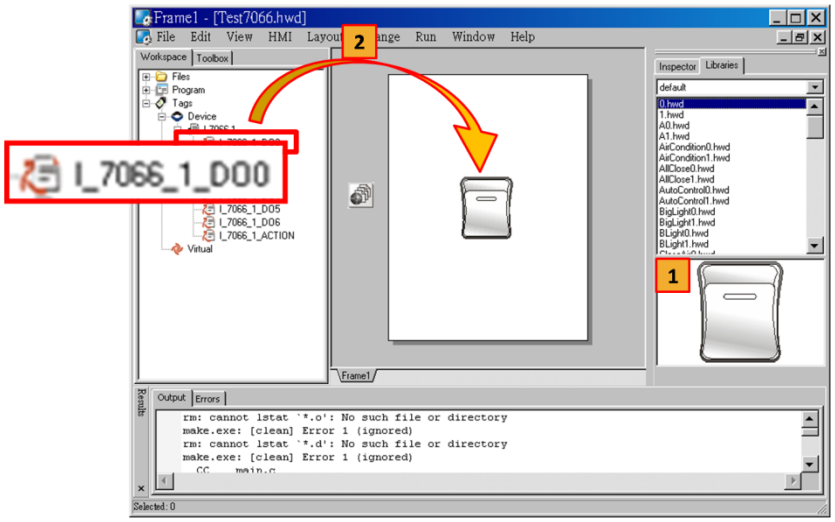


properties	description
Height	Vertical side length of the ObjectList
ID	The serial number of tools (including Drawings and Widgets) which is used to identify them.
Left	x-coordinate of the left-top vertex of the ObjectList
Name	The name of the ObjectList
Objects	The maintained library objects
Top	y-coordinate of the left-top vertex of the ObjectList

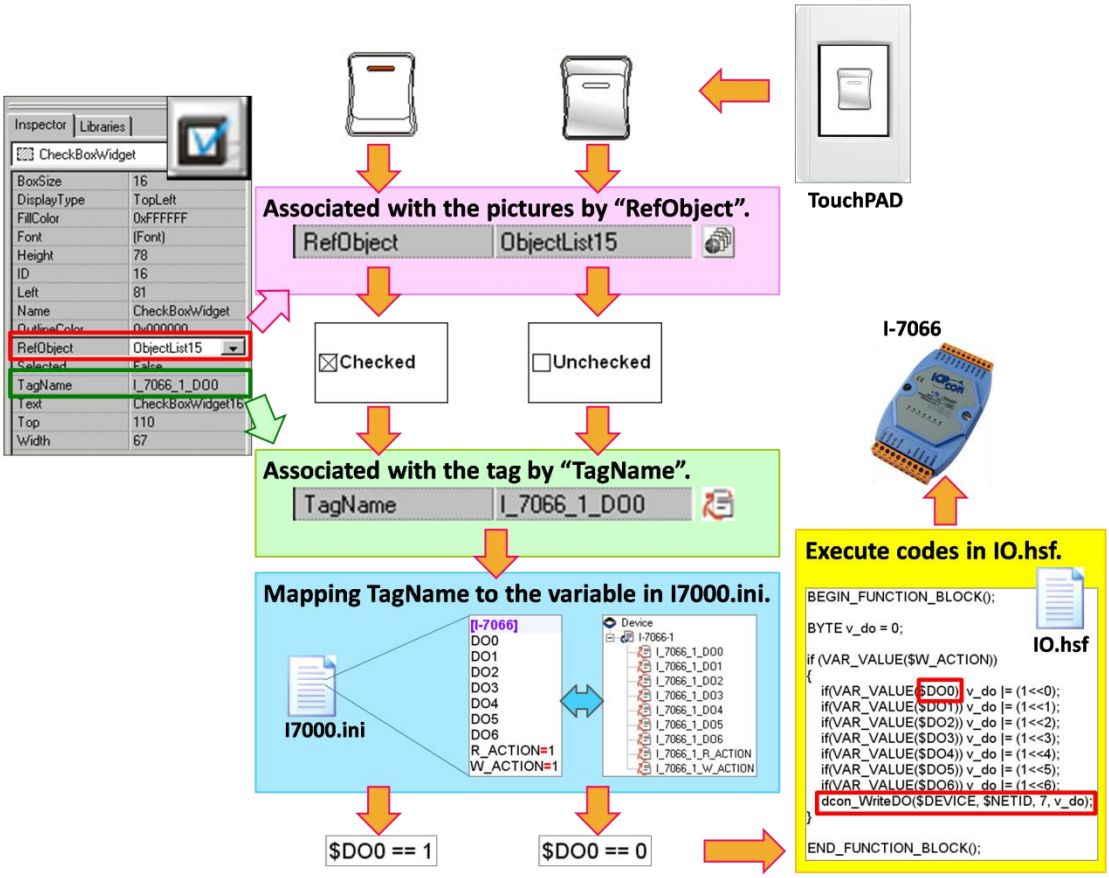
Width	Horizontal side length of the ObjectList

Relationships between TouchPAD and I/O module

Take I-7066 for example, **Register Devices (F3)** to automatically generate tags and then drag and drop the tag on the frame.



HMIWorks does the followings to build the relationships between TouchPAD and I/O modules.



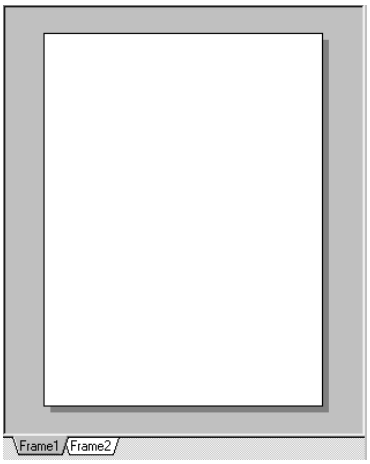
Note: the TagName property takes effect only in the programming type, Ladder. (It's easier in programming type "Standard C". Control the I/O by using API function, dcon_WriteDO, in the event handler of the CheckBox.)

5.5. Menus

All the menus can be accessed from menu bar or the popup menu.

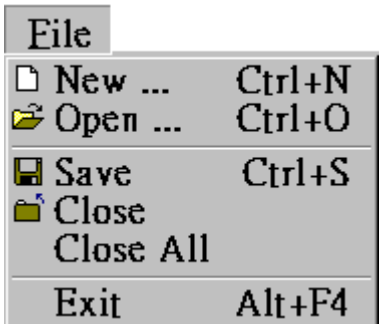
The menu bar:

Right click on the frame design space, popup menu shows.
The frame design space:



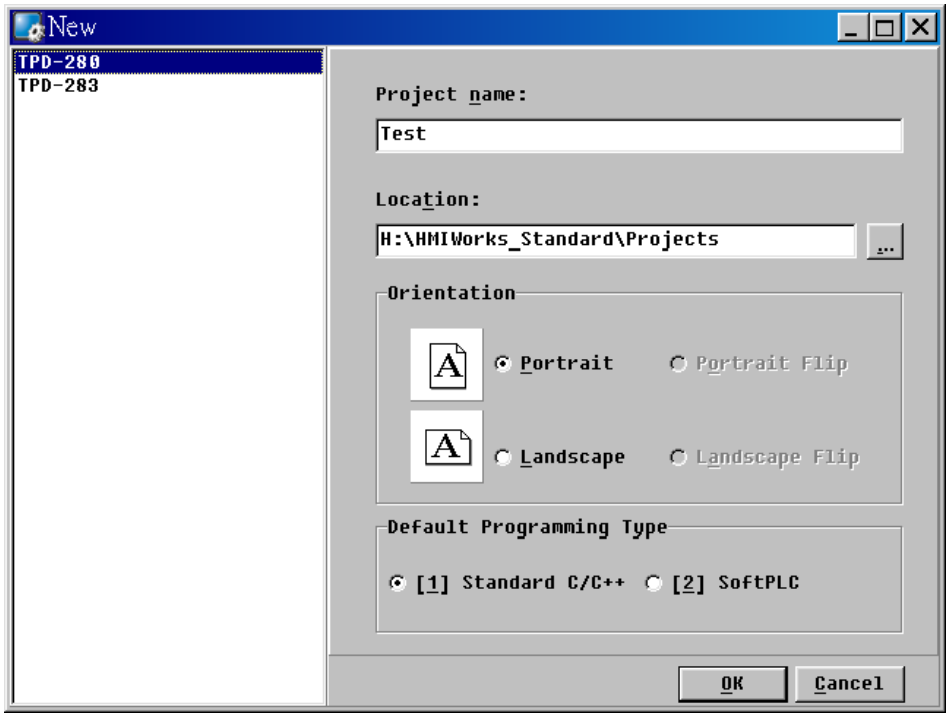
Note:
The items in the HMI menu are introduced in the section 4.3 **Ladder Designer**.

5.5.1. Starting Design, File Menu



New …

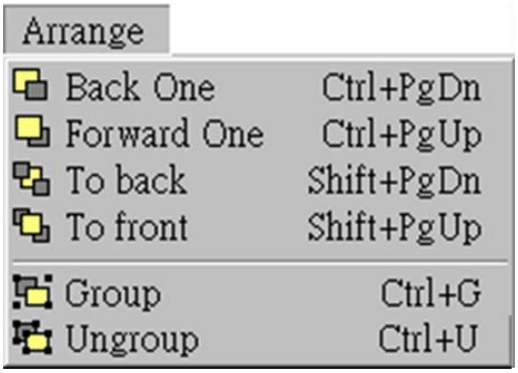
Create a new project.
Specify project name, its location, the orientation, the programming type and select the module model.
Then press **OK**.



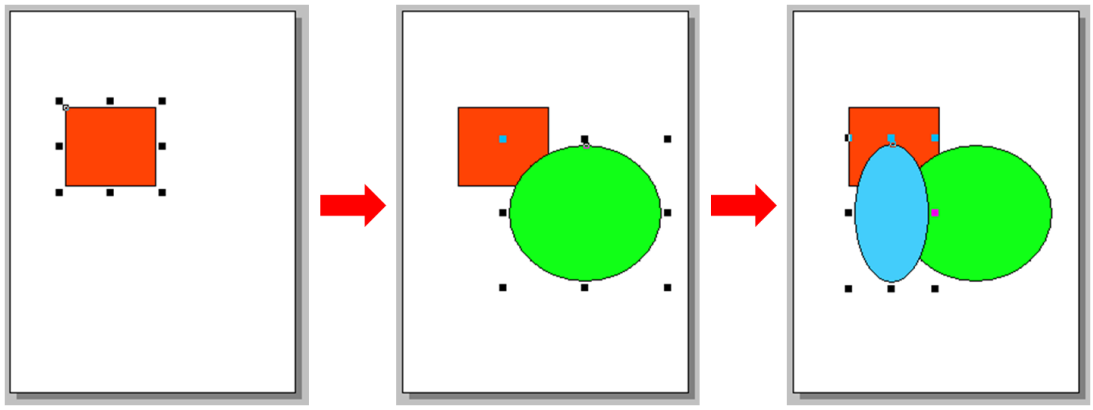
Open …

Open a HMIWorks project (.hxp).

5.5.2. Cascading and Grouping, Arrange Menu

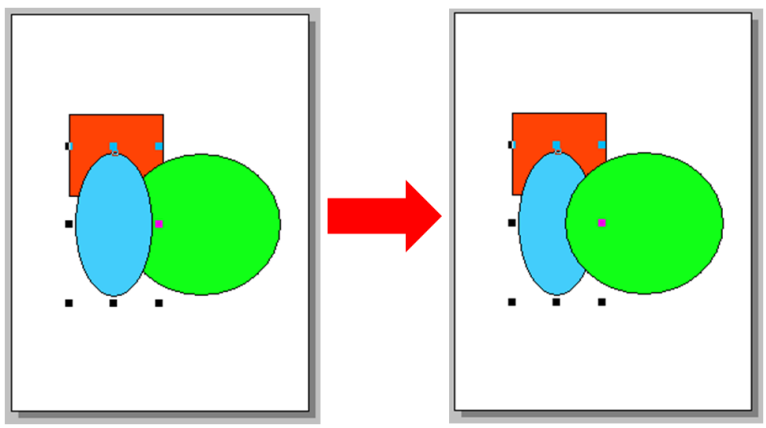


To demonstrate functions of cascading and grouping, first draw three shapes as followings:



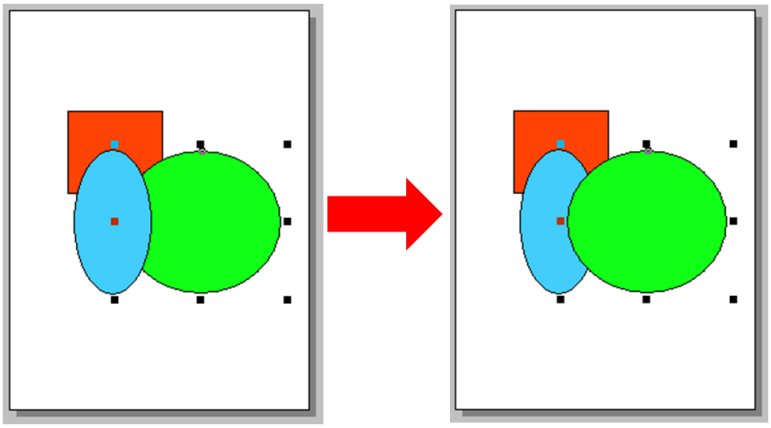
Back One

Make the selected object go down a level of the stacks.
For example, select the blue ellipse and click "**Back One**" in the menu.
You can see that the blue ellipse goes down one level in the stack.



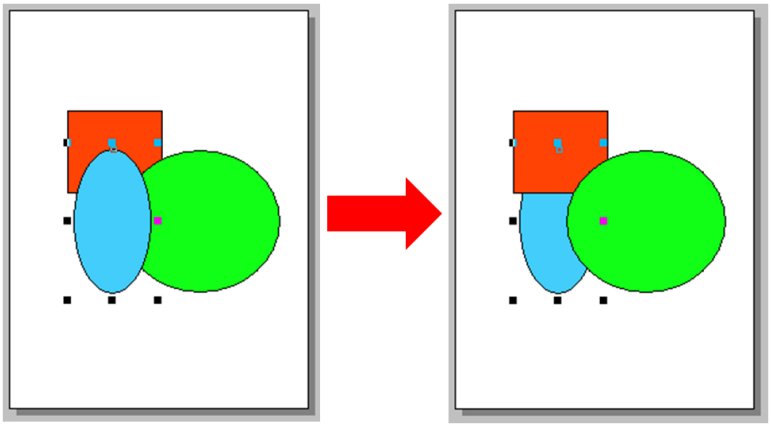
Forward One

Make the selected object go up a level of the stack.
For example, select the green circle and click “**Forward One**” in the menu. You can see that the green circle goes up one level in the stack.



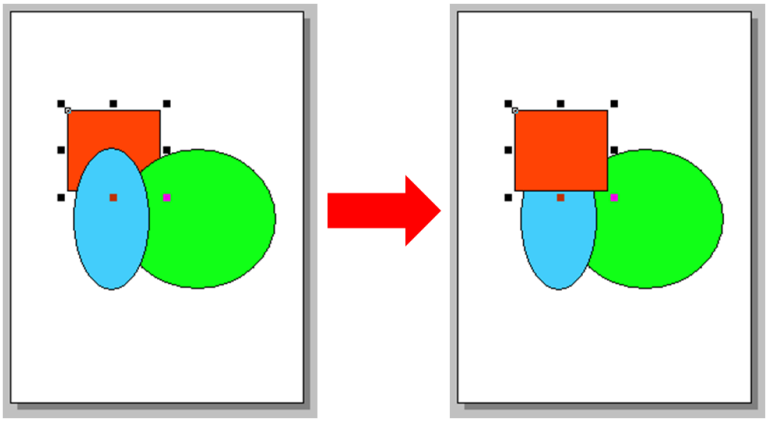
To back

Make the selected object go down to the lowest level of the stack.
For example, select the blue ellipse and click “**To back**” in the menu. You can see that the blue ellipse goes down to the lowest level in the stack.



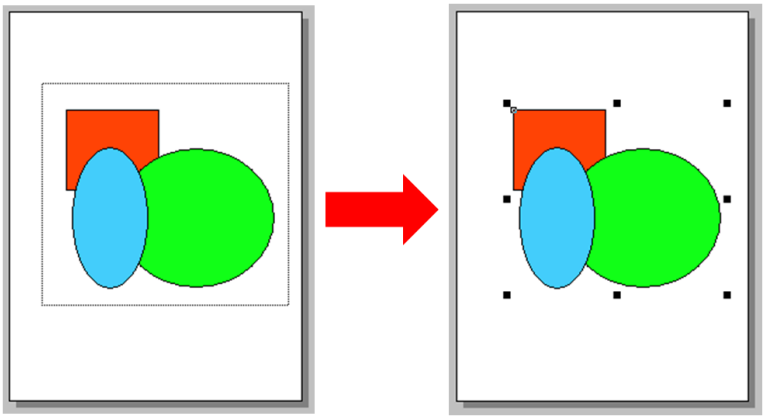
To front

Make the selected object go up to the highest level of the stack.
For example, select the red square and click “**To front**” in the menu. You can see that the red square goes up to the highest level in the stack.



Group

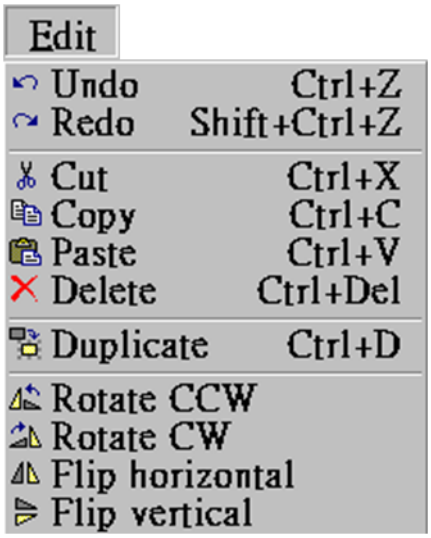
Put tools(Drawing tools, Widgets and System components) together as a set, that is, a group.
For example, first circle the items together by a mouse, and then click "**Group**" in the menu. You can see that they are grouped together.



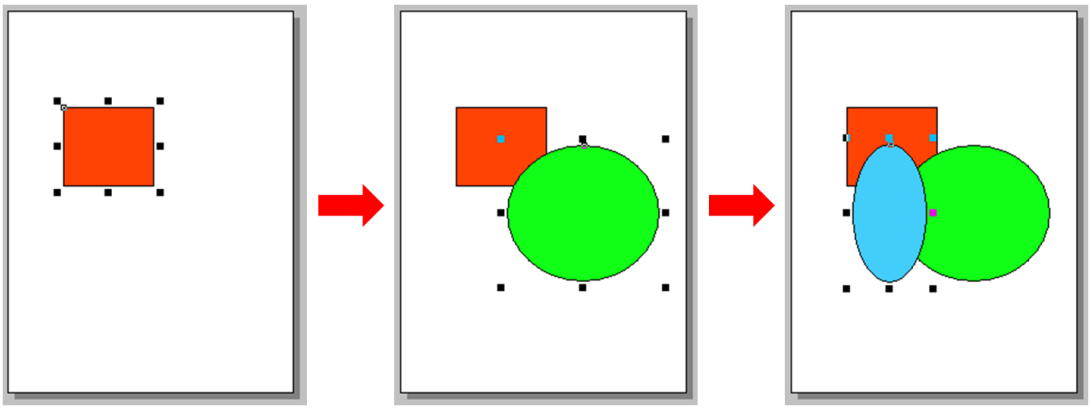
Ungroup

Break a group into its original tools.
For example, select the group and then click "**Ungroup**" in the menu.

5.5.3. Rotating and Flipping, Edit Menu

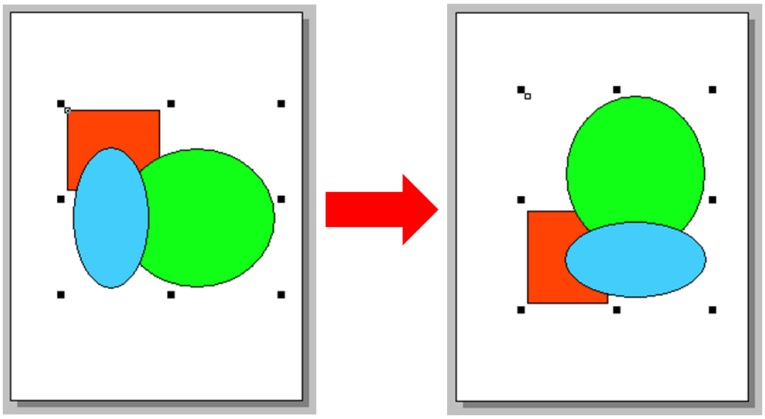


To demonstrate functions of rotating and flipping, first draw three shapes as followings:



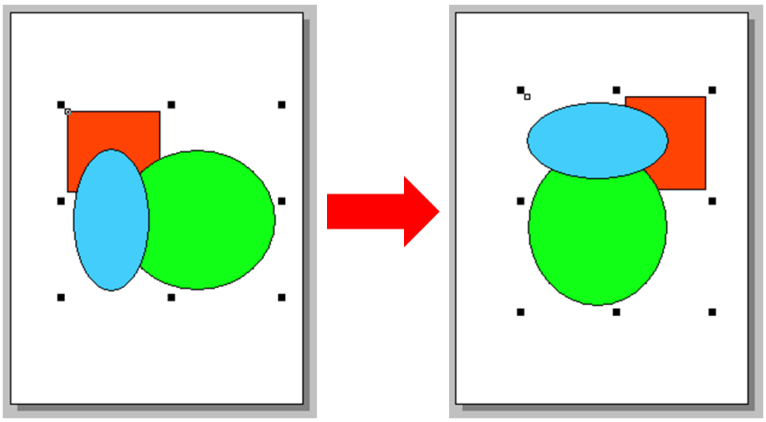
Rotate CCW

Rotate the selected item in the counter-clockwise direction.
For example, first put three items into one group, select the group and then click on "**Rotate CCW**" in the menu. You can see that this group of shapes is rotated counter-clockwise.



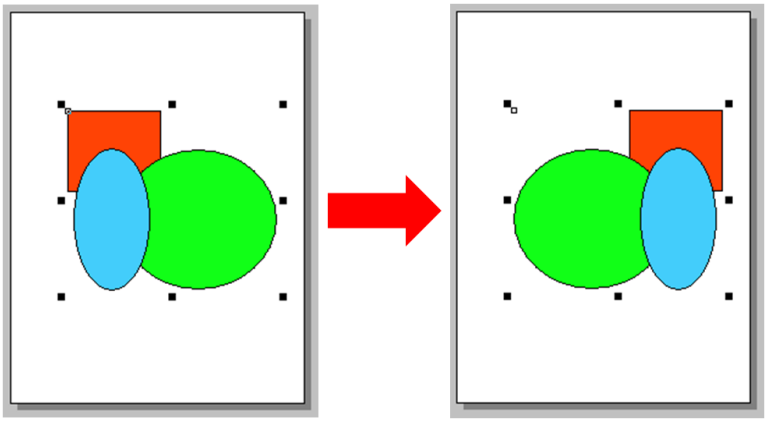
Rotate CW

Rotate the selected item in the clockwise direction
For example, first put three items into one group, select the group and then click on “**Rotate CW**” in the menu. You can see that this group of shapes is rotated clockwise.



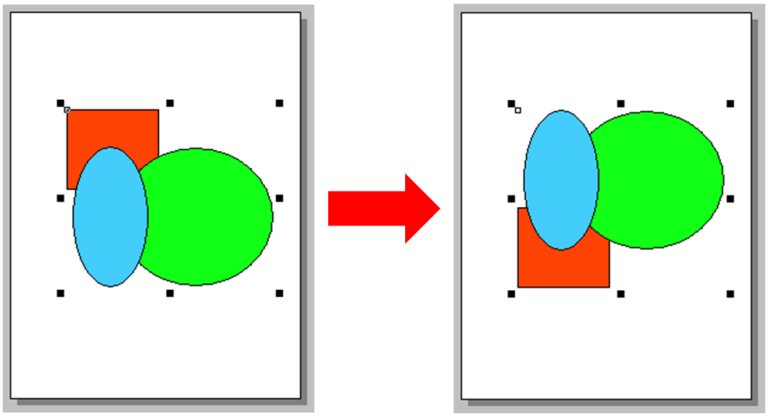
Flip horizontal

Flip the selected item in the horizontal direction.
For example, first put three items into one group, select the group and then click on “**Flip horizontal**” in the menu. You can see that this group of shapes is flipped horizontally.

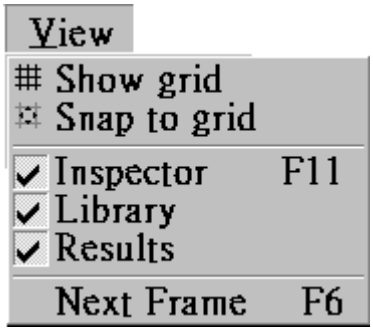


Flip vertical

Flip the selected item in the vertical direction.
For example, first put three items into one group, select the group and then click on "**Flip vertical**" in the menu. You can see that this group of shapes is flipped vertically.

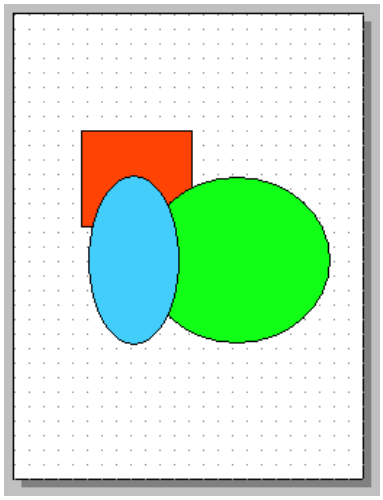


5.5.4. Gridding and Viewing, View Menu



Show grid

Show grid on the frame. Take the following snapshot for example, and then you can see the dots on the frame. Those dots represent the grid.



Snap to grid

Snap graphics or widgets to the grid.

Inspector

Display the **inspector** window or not

Library

Display the **library** window or not

Results

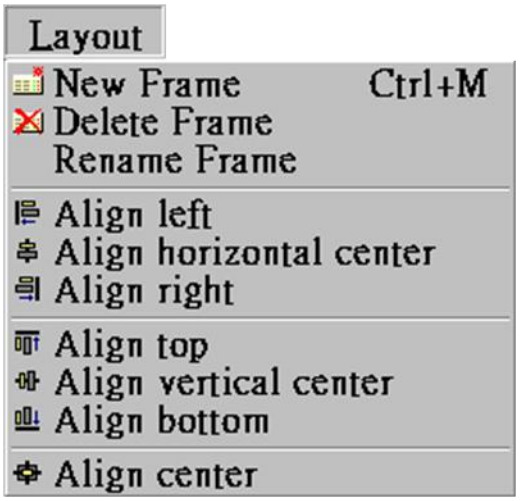
Display the **Results** window or not

Next Frame

Go to next frame

5.5.5. Frame Managing and Aligning, Layout

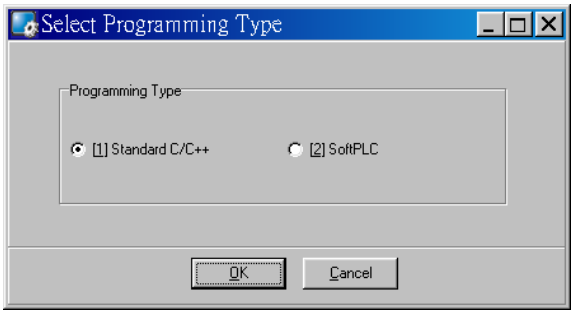
Menu



- **Frame Management:**

New Frame

Create a new frame (select the programming type)



Delete Frame

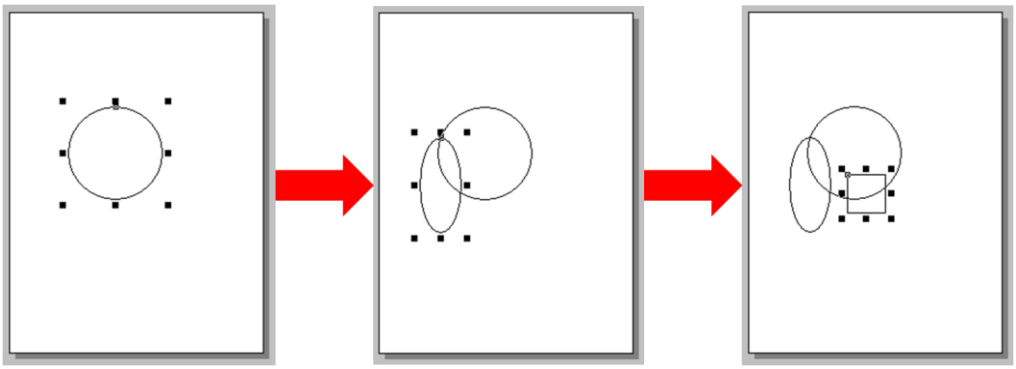
Delete the current frame

Rename Frame

Rename a frame

- **Alignment:**

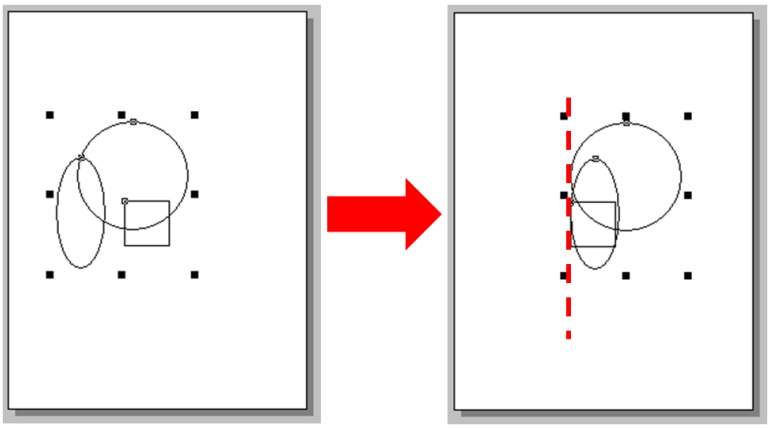
To demonstrate the functions of alignment, draw three shapes as followings



Note: all alignment functions refer to the last shape you draw. In above example, all alignment functions refer to the square.

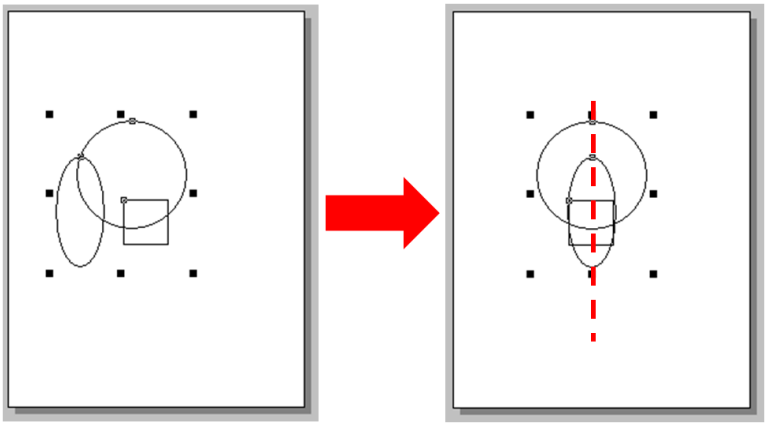
Align left

Align the leftmost edge of all the selected items to that of last-drawn item.
For example, select all the items and then click "**Align left**" in the menu.



Align horizontal center

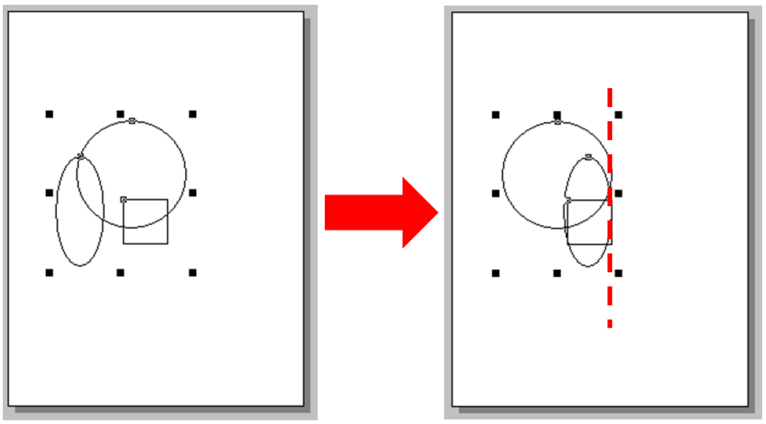
Align the horizontal center of all the selected items to that of last-drawn item.
For example, select all the items and then click "**Align horizontal center**" in the menu.



Align right

Align the rightmost edge of all the selected items to that of last-drawn item.

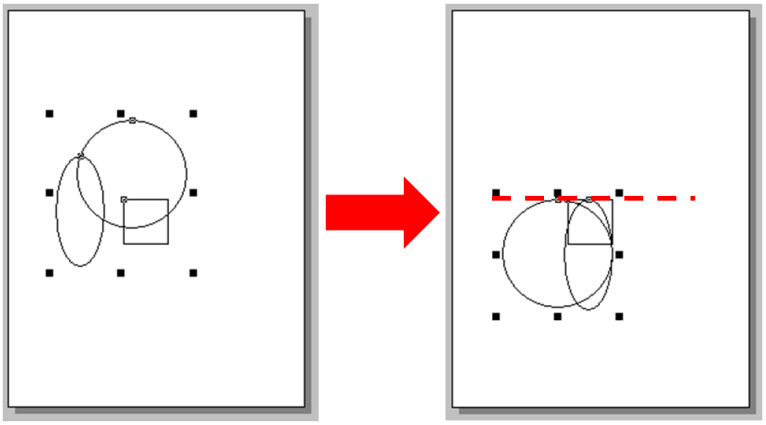
For example, select all the items and then click “**Align right**” in the menu.



Align top

Align the topmost edge of all the selected items to that of last-drawn item.

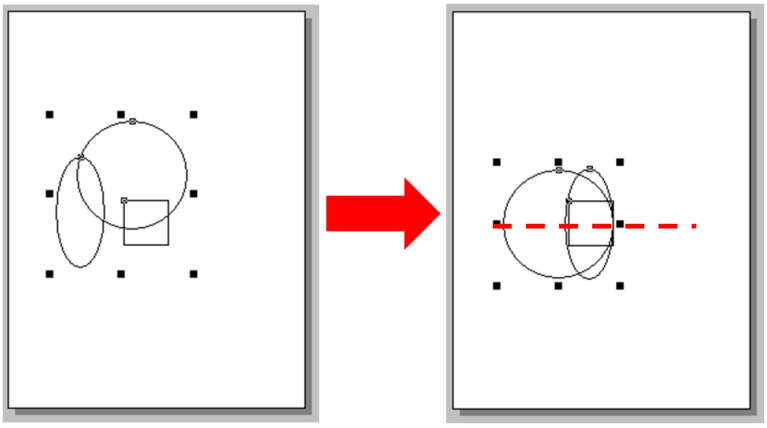
For example, select all the items and then click “**Align top**” in the menu.



Align vertical center

Align the vertical center of all the selected items to that of last-drawn item.

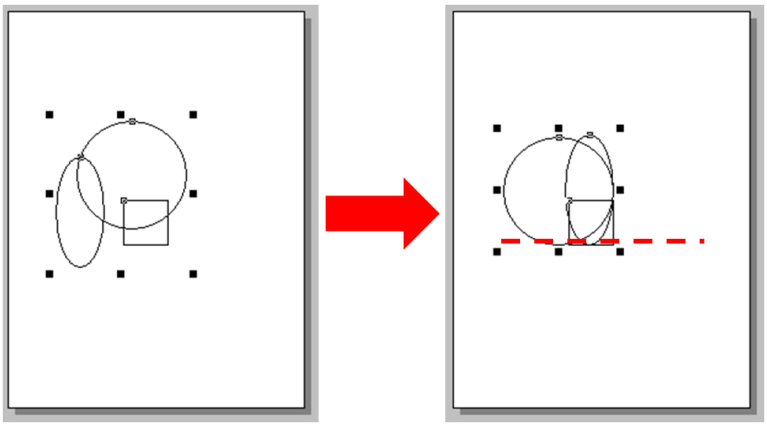
For example, select all the items and then click “**Align vertical center**” in the menu.



Align bottom

Align the bottommost edge of all the selected items to that of last-drawn item.

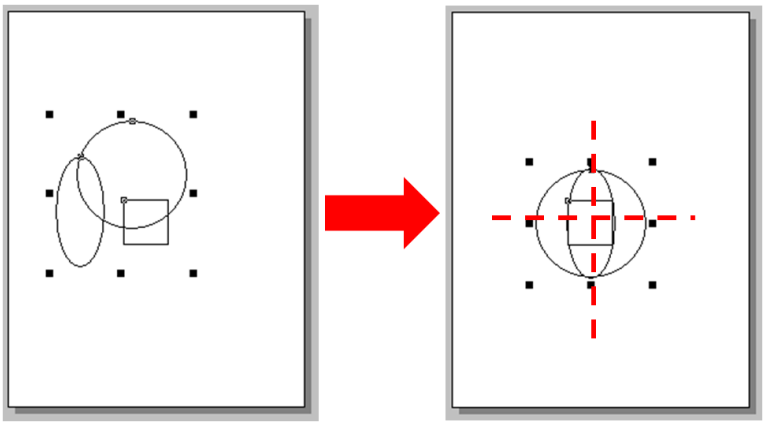
For example, select all the items and then click “**Align bottom**” in the menu.



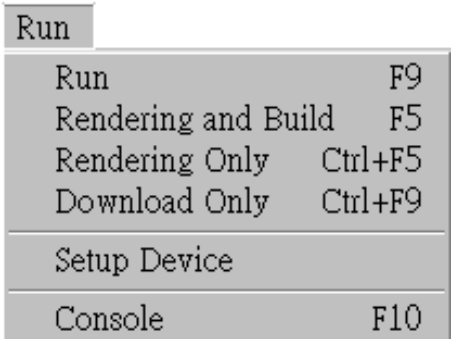
Align center

Align the center point of all the selected items to that of last-drawn item.

For example, select all the items and then click “**Align center**” in the menu.



5.5.6. Build and Download to Run, Run Menu



Refer to the chapter, “Setup Devices and Connect to I/O”, for “Setup Device”.

Other Items in the Run Menu

Run (F9)

➔ Rendering + Build + Download

Rendering and build (F5)

➔ Rendering + Build (Compile and Link)

Rendering Only (Ctrl + F5)

➔ Generate source codes for frames, tools, ladders, etc.

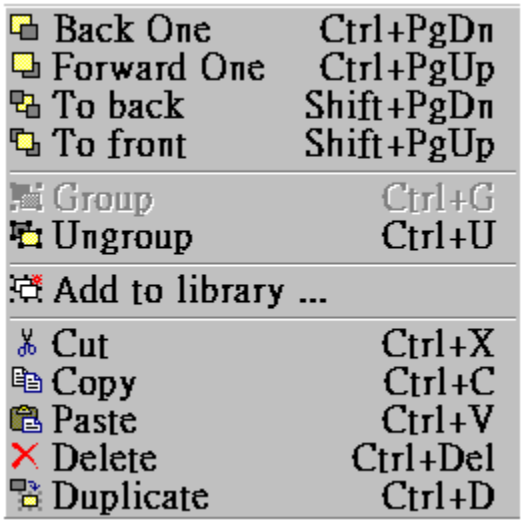
Download Only (Ctrl + F9)

➔ Download building bin file to TouchPAD

Console (F10)

➔ Evoke console window (cmd.exe) with environment variables for HMIWorks. Users can modify the generated codes and then re-make the codes. (Use make.exe.)

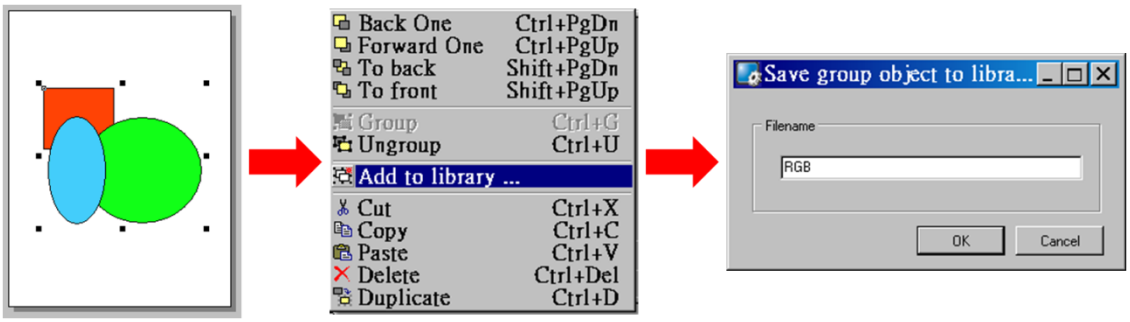
5.5.7. Library Management, Popup Menu



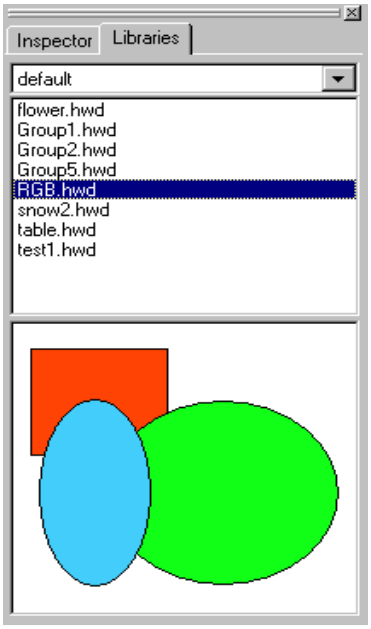
How to add items to library

For example as below:

1. Group the selected items.
2. Right click on the frame panel to evoke the popup menu.
3. Click on "**Add to library …**"
4. Specify the library name and save the library.



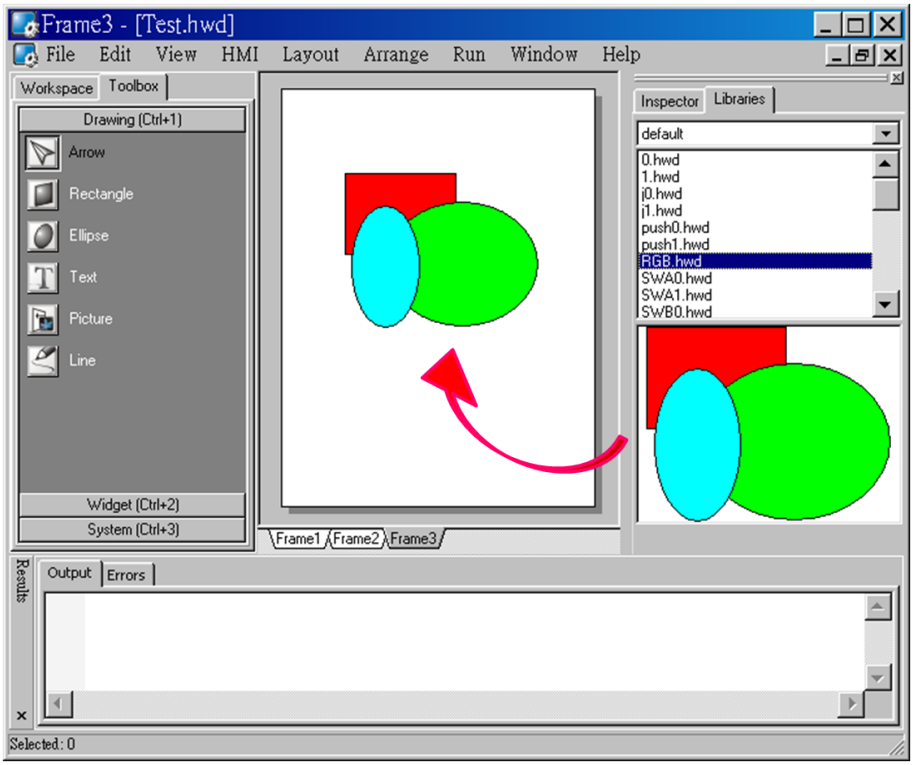
Note: you can preview the library in the **library** window.



How to use items from library

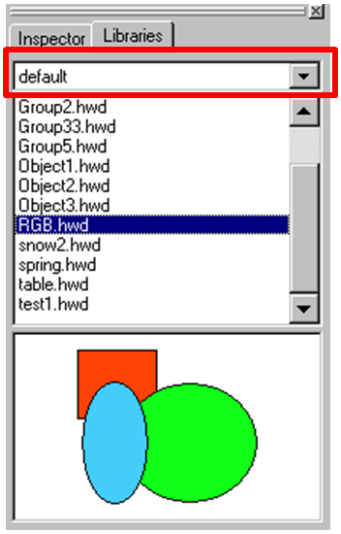
For example as below:

1. Click on the tab of "**Libraries**" to show library window.
2. Pick the object you want. You can preview the object in the preview box.
3. Click (and not released) on the item in the preview box and then drag the preview item and drop it on the frame panel.



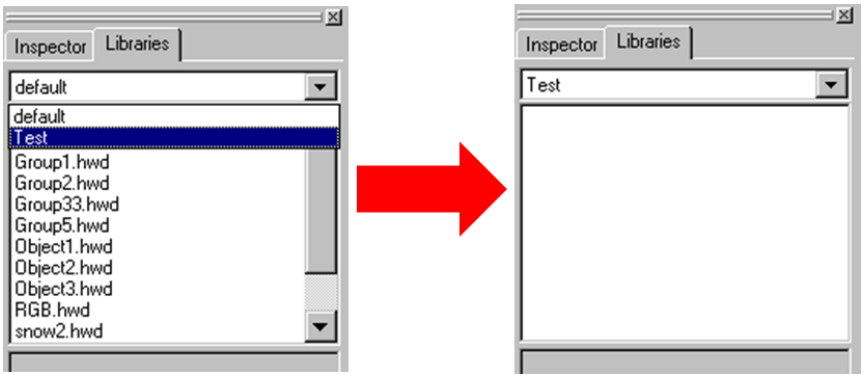
How to add a new folder into library window

By default we have the only folder “default” in the library window as the below shows.



To add a new folder into the **library** window, create a new folder in the following path:
“HMIWorks_install_path\bin\Lib\” where the HMIWorks_install_path is the installation path of HMIWorks.

Supposed the installation path of HMIWorks is “C:\ICPDAS\HMIWorks_Standard”. And we want to add a new folder named “Test” into the **library** window. Then all we have the do is creating a new folder named Test in the directory of “C:\ICPDAS\HMIWorks_Standard\bin\Lib”.
And then reopen the **library** window, you can see the new folder “Test” as below shows.
Of course, there’s no library item in it. You should add items yourself.



6. Make a Simple Project

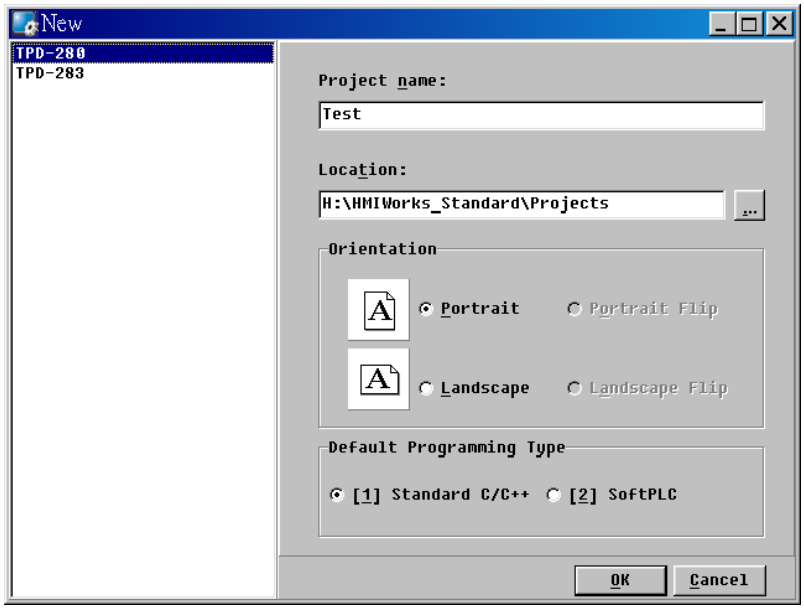
There are two programming types in HMIWorks. In this chapter, we introduce how to build your first project for each programming type.

6.1. Your First Project Using Standard C

1. Create a new project

Go to the menu: **File → New…**, and select the Model, then specify the Project name, the Location, the Orientation, and the Programming Type.

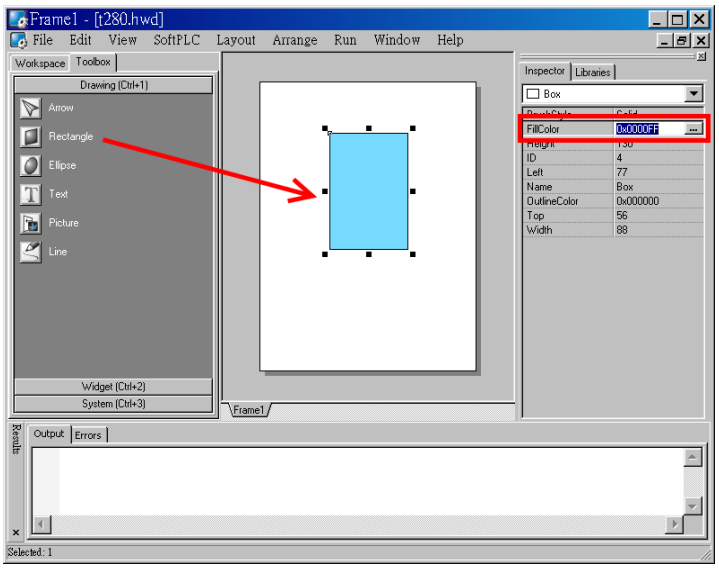
Here we choose **programming type** as [1] Standard C.



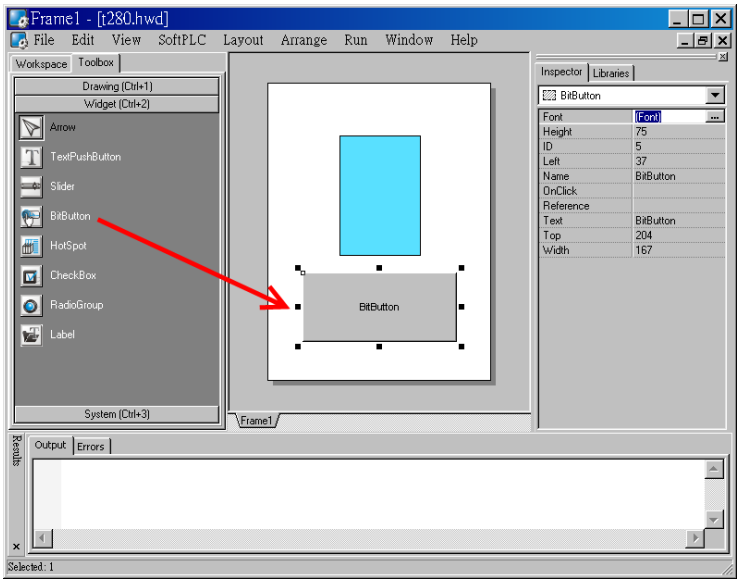
2. Design the Graphic User Interface

For example, draw a rectangular and fill the color. Of course, you can draw more complex and beautiful figures. Here, we simply

demonstrate how to make a simple project.

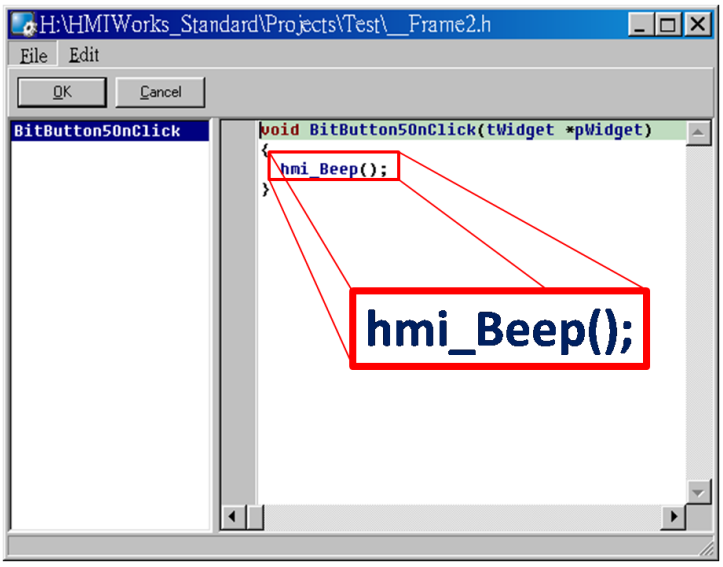


And then select a Widget. For example, pick a BitButton.



3. Modify Source Codes

Double click on the BitButton to invoke the code-editing window.
Use “hmi_Beep();” to sound a beep for example, then press **OK**.
Of course, more codes can be written to fulfill the requirements. Here we simply demonstrate how to make a simple project so we make the codes as simple as possible.

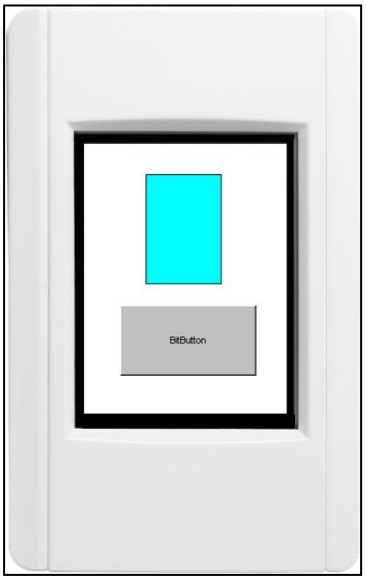


4. Setup Device

Refer to “Setup Devices” in the chapter 4 for details.

5. Compile and Download to Run

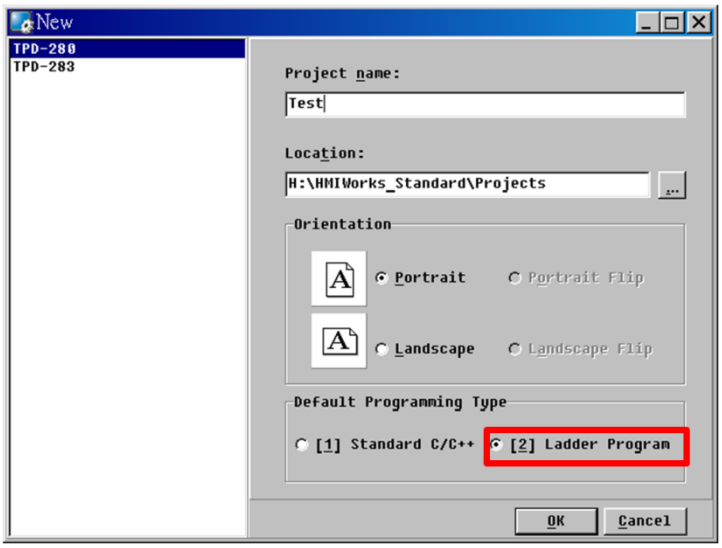
After connecting to TouchPAD, press **F9** to run (or **Run → Run**).
As the following figure shows, pressing the button makes TouchPAD sound a beep.



6.2. Your First Project Using Ladder

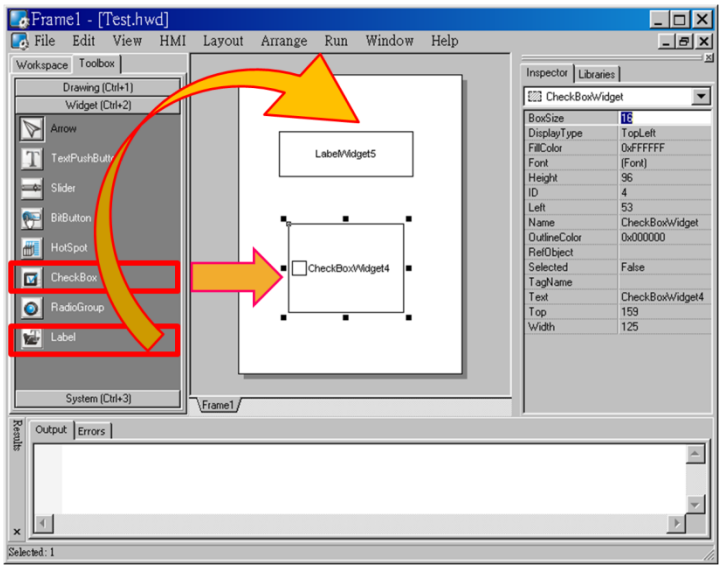
1. Create a new project

Go to the menu: **File → New…**, and select the Model, then specify the Project name, the Location, the Orientation, and the Programming Type. Here we choose **programming type** as [2] Ladder.

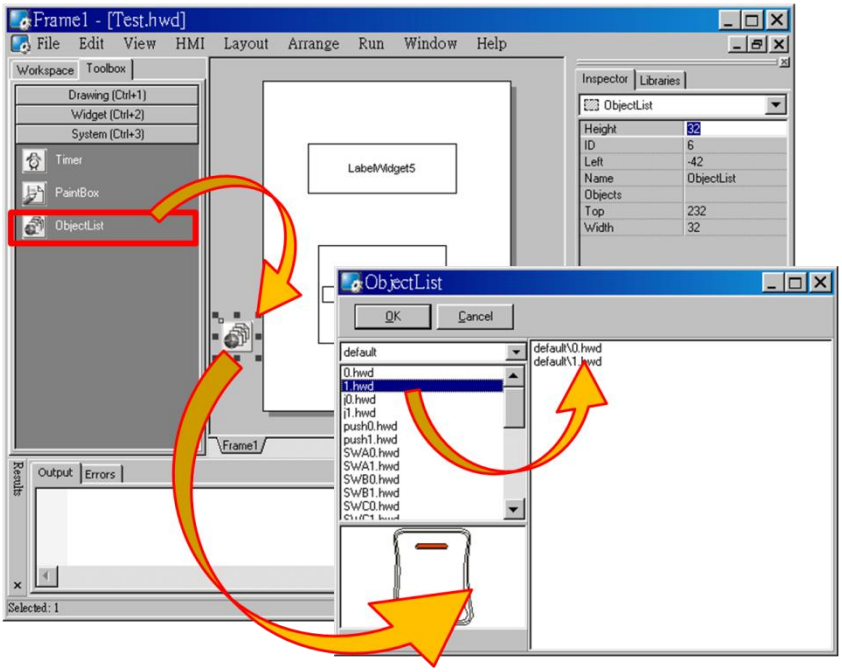


2. Design the Graphic User Interface

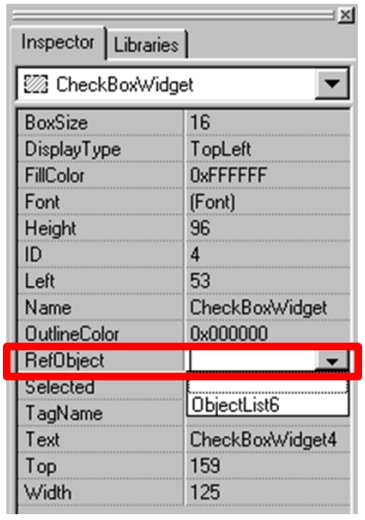
For example, place a CheckBox and a Label on the frame. Here, we plan to take the CheckBox as an input and the Label as an output.



Drag an ObjectList and drop in the frame design space. Double click on the ObjectList icon to evoke the "**ObjectList**" window. In the "**ObjectList**" window, double click on the pictures you want. Users need to double click on two pictures, one is for the checked state of the CheckBox and the other is for the unchecked state. Press **OK** to finish this step.

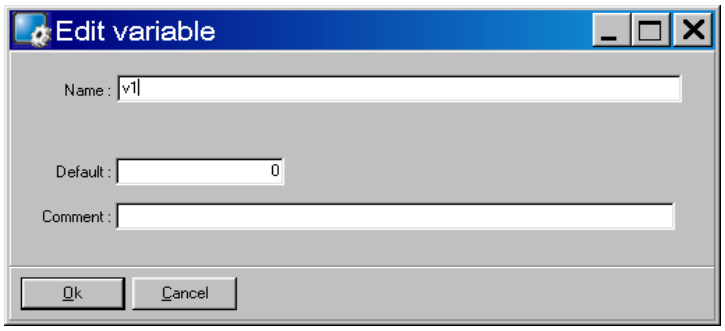


Make the CheckBox refer to the ObjectList by setting the property "RefObject" to the ObjectList. Now toggling the states of the CheckBox becomes the exchanges of the pictures in the ObjectList.

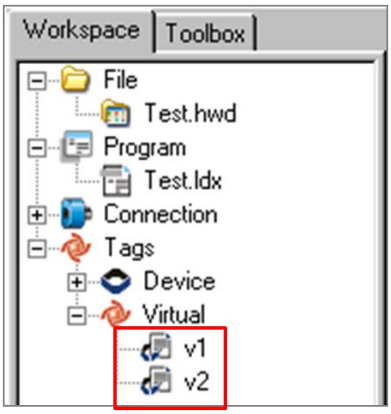


3. Design the Ladder Diagram

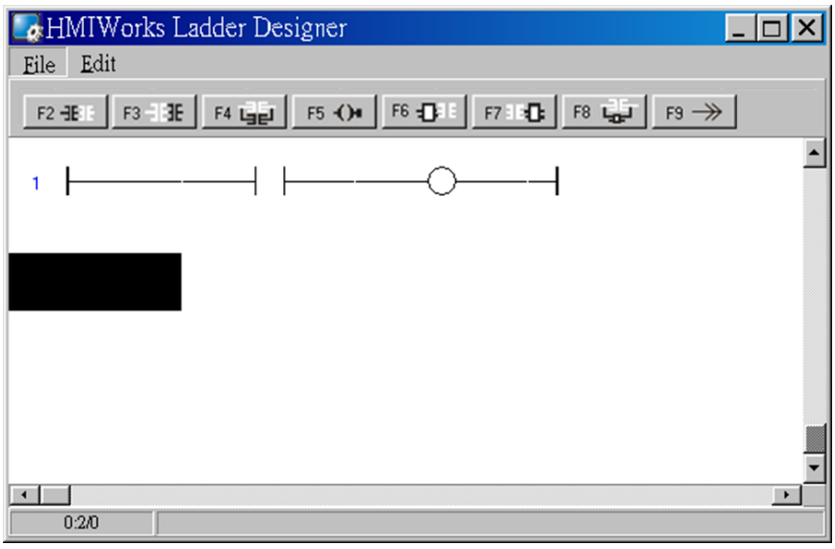
First, add virtual tags (variables) for the ladder diagram. Press **F2** key or click on "**New Virtual Tag**" in the "**HMI**" menu. Here, we add two tags, v1 and v2, for example.



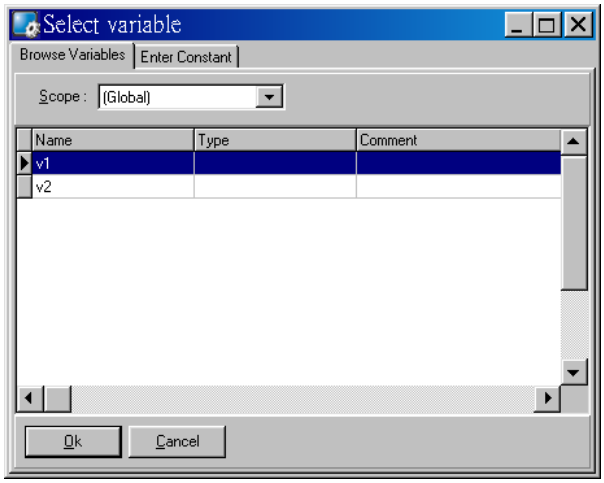
After adding the tags, users can check in the **Workspace**.



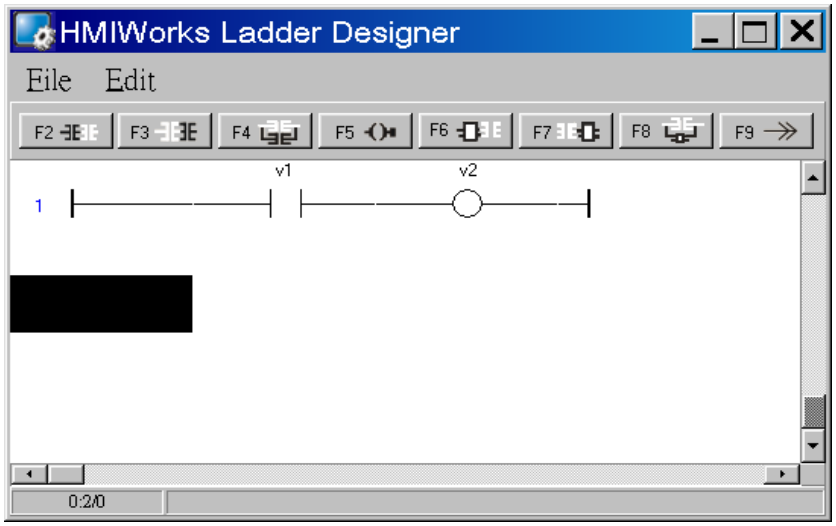
Press **F4** key or click on "**Ladder Designer**" in the "**HMI**" menu to open the **Ladder Designer**. In the **Ladder Designer**, press **F2** to create a new rung.



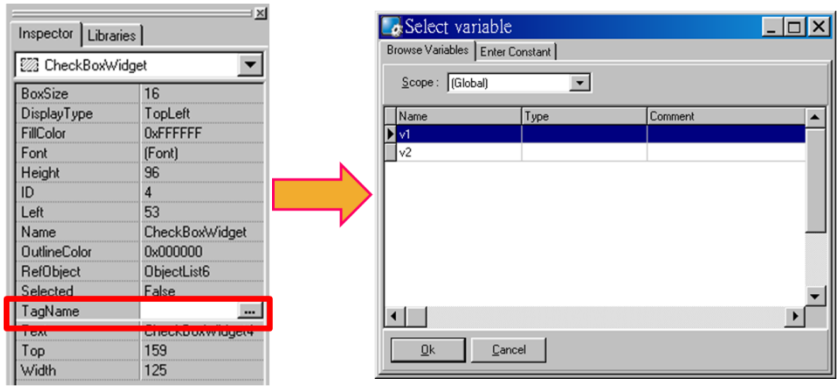
Double click on the contact input of the first rung in the **Ladder Designer**. Then a "**Select variable**" shows. Choose the variable to associate with the contact input.



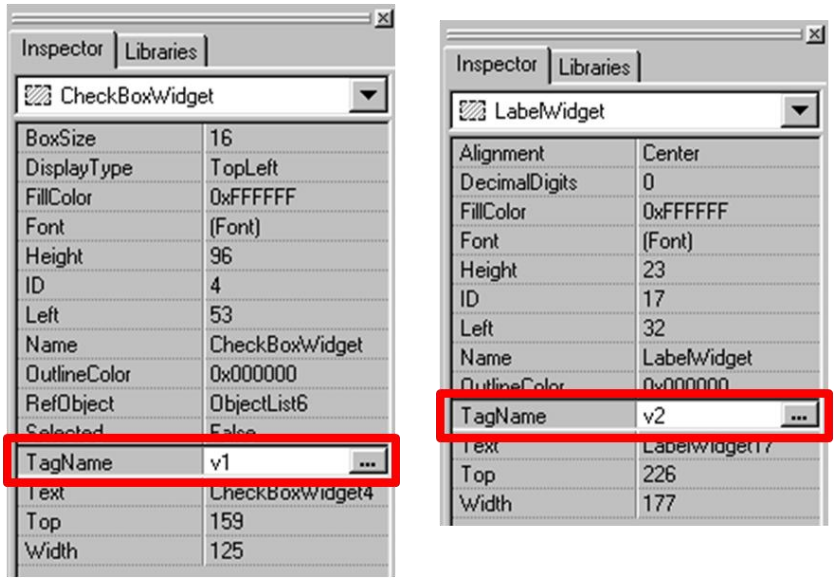
Here, we select variable v1 to associate the contact input. And then we associate v2 with the coil output in the similar way.



Then we associate the CheckBox with v1 and the Label with v2 by "TagName" properties of themselves.



After setting the "TagName" properties, users can check in the **Inspector**.

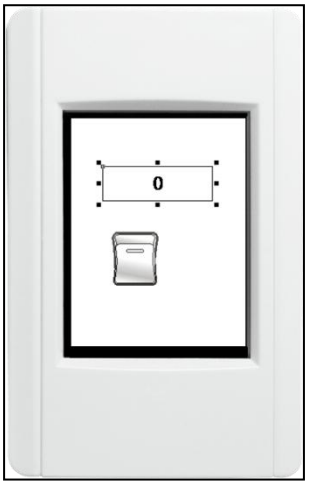


4. Setup Device

Refer to “Setup Devices” in the chapter 4 for details.

5. Compile and Download to Run

After connecting to TouchPAD, press **F9** to run (or **Run → Run**).
As the following figure shows, pressing the button changes the value of the Label from 0 → 1, or 1 → 0.

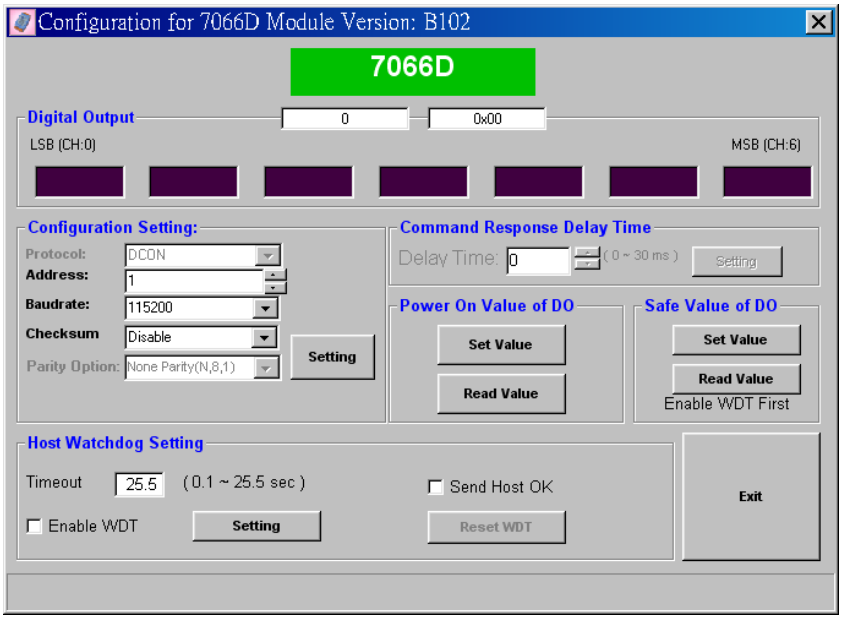


6.3. Integrate TPD-280 Series with I/O modules

In this example, we use TPD-280 to control an I-7066, the 7-channel photo-MOS relay output module of ICP DAS. First, put I-7066 in the same RS-485 network of TPD-280 and configure I-7066 with DCON Utility (baudrate, data bit, parity, stop bit, Net ID).

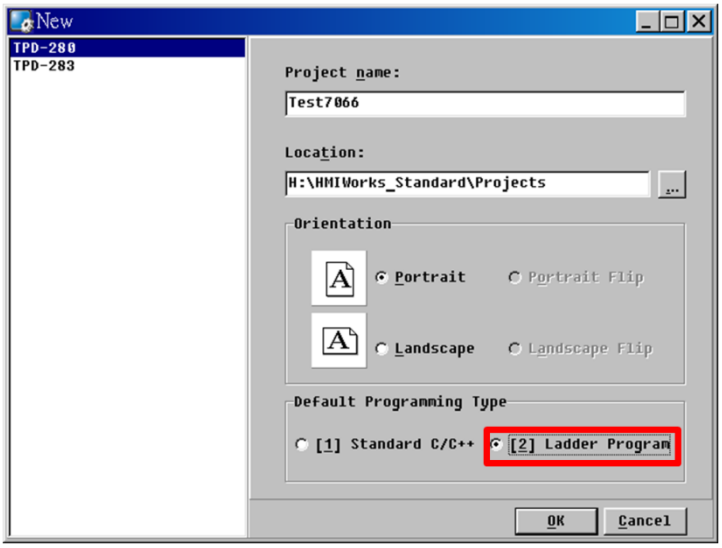
1. Use DCON Utility to setup I-7066

Download DCON Utility to install and refer to the user manual.
ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/driver/dcon_utility/
Then use DCON Utility to setup I-7066. Mainly set the followings: Address(Net ID), Baudrate, Databit, Parity, Stopbit.



2. Create a new project

Go to the menu: **File → New…**, and select the Model, then specify the Project name, the Location, the Orientation, and the Programming Type. Here we choose **programming type** as [2] Ladder.



3. Design the Graphic User Interface

We can skip this step.

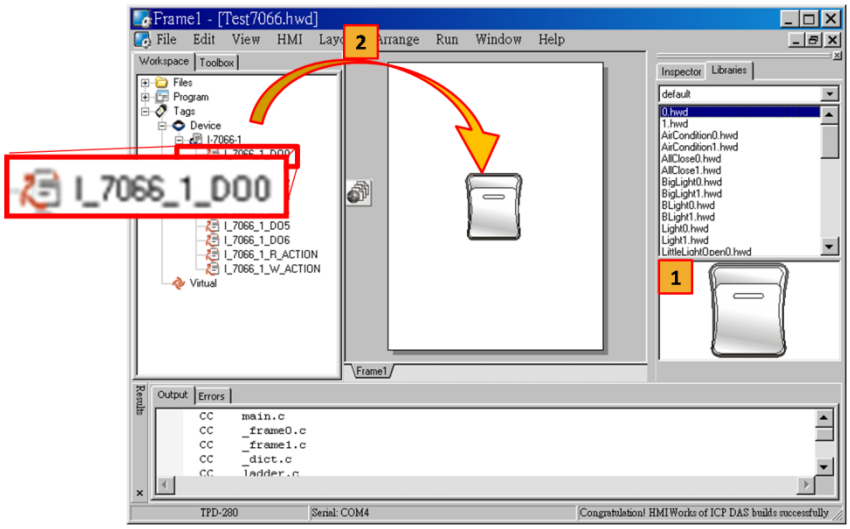
Here we just demonstrate how to quickly complete a whole new project with I/O modules of ICP DAS.

4. Design the Ladder Diagram

Press **F3** key or click on "**Register Devices**" in the "**HMI**" menu to evoke the "**Devices**" window to register I-7066.

Refer to section "Connect to I/O Modules" in the chapter 4 for details.

Then pick up the picture which represents the tag. Drag and drop the tag that is corresponding to the DO0 of I-7066. On the frame design space, the picture you just pick is now on the frame.



5. Setup Device

Refer to "Setup Devices" in the chapter 4 for details.

6. Compile and Download to Run

After connecting to TouchPAD, press **F9** to run (or **Run → Run**). As the following figure shows, pressing the button changes the output of channel 0 of the I-7066.

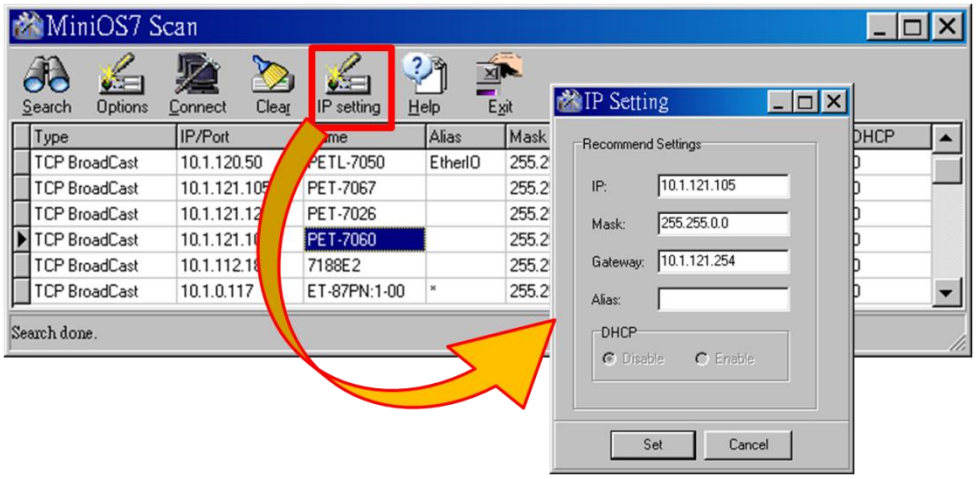
6.4. Integrate TPD-283 Series with I/O modules

In this example, we use TPD-283 to control a PET-7060, the 6-channel Power Relay Output and 6-channel Isolation Digital Input with PoE module of ICP DAS. First, put PET-7060 in the Ethernet network of TPD-283 and use browser to configure PET-7060.

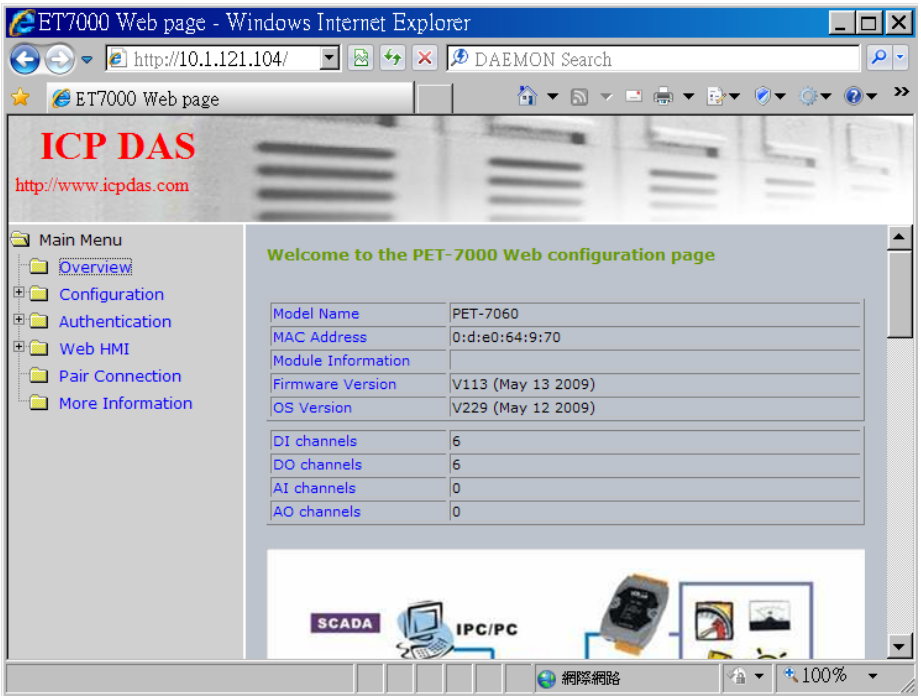
1. Configure PET-7066 Using a Browser

Download MiniOS7 Utility and its user manual on ftp://ftp.icpdas.com/pub/cd/8000cd/napdos/minios7/utility/minios7_utility/

Then use MiniOS7 Utility to change the IP setting of PET-7060. (Be sure to make PET-7060 and your PC in the same subnet.) Press **F12** to scan the network. After searching, click on PET-7060 and then click on "**IP setting**" button to set IP.

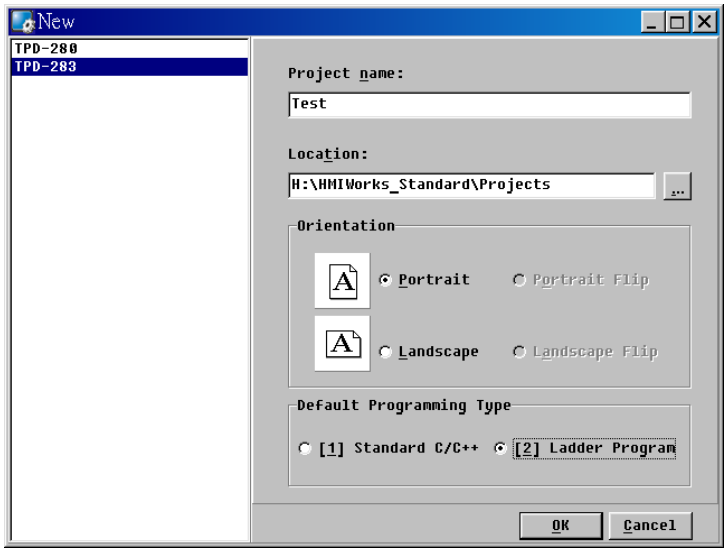


Finally, use a browser to connect to the PET-7060 and configure it.



2. Create a New Project

Go to the menu: **File → New…**, and select the Model, then specify the Project name, the Location, the Orientation, and the Programming Type. Here we choose **programming type** as [2] Ladder.



3. Design the Graphic User Interface

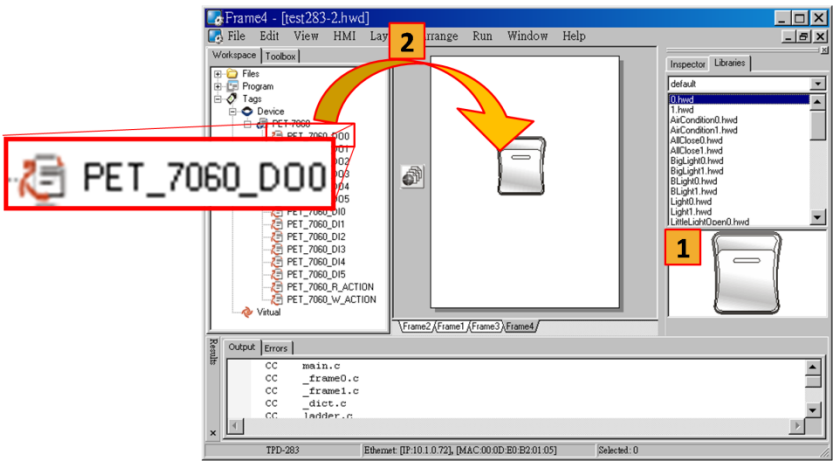
We can skip this step.
Here we just demonstrate how to quickly complete a whole new project with I/O modules of ICP DAS.

4. Design the Ladder Diagram

Press **F3** key or click on "**Register Devices**" of the "**HMI**" menu to evoke the "**Devices**" window to register PET-7060.

Refer to section "Connect to I/O Modules" in the chapter 4 for details.

Then pick up the picture which represents the tag. Drag and drop the tag that is corresponding to the DO0 of PET-7060 to the frame design space. On the frame design space, the picture you just pick is now on the frame.



5. Setup Device

Refer to “Setup Devices” in the chapter 4 for details.

6. Compile and Download to Run

After connecting to TPD-283, press **F9** to run (or **Run** → **Run**).
As the following figure shows, pressing the button changes the output of channel 0 of the PET-7060.

7. Advanced Programming in C

We have an API reference for TouchPAD. However, the API reference does not include API functions for handling widgets, such as TextPushButton, CheckBox, etc.

Though you can refer to the generated codes to learn how to use these API functions, all the API functions are defined in header files in the following path: “C:\ICPDAS\HMIWorks_Standard\include\grlib” and “C:\ICPDAS\HMIWorks_Standard\include”, where “C:\ICPDAS\HMIWorks_Standard” is the installation path.

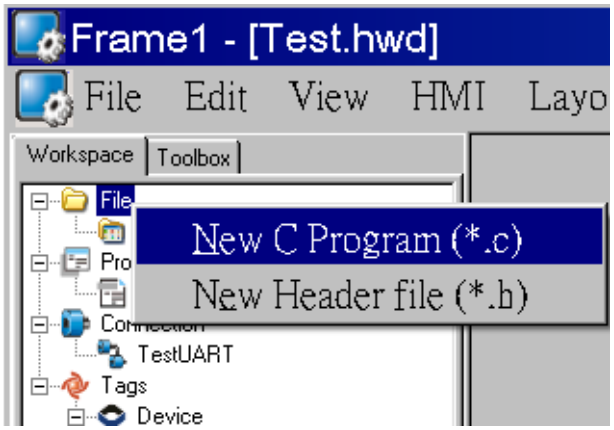
We give some examples in this chapter.

7.1. Add a New File to Project

Before introducing the details, first we show how to add a new file (.c or .h) to the project.

1. Go to **Workspace**.
2. Right click on the item “File” and a pop-up menu shows.
3. On that pop-up menu, choose the type of the file you want to add.

As the following figure shows:

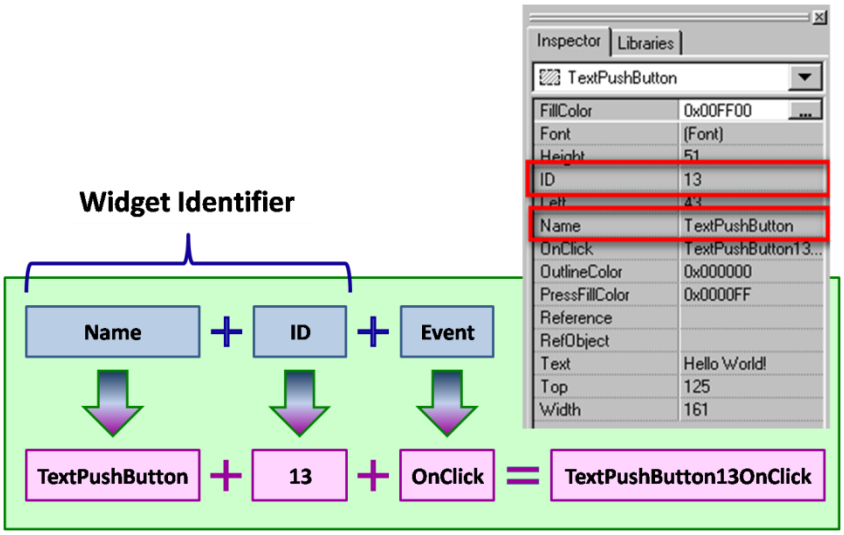


7.2. Update Properties in Run Time

It is a bit more complicated to change the properties of widgets in the run time. In this section, we demonstrate some commonly-used cases, including:

1. FillColor and Text of a TextPushButton
2. Percentage of a Slider
3. Selected of a CheckBox
4. Font, Text and TextColor of a Label

Updating properties is implemented in the event handlers of widgets.
Note that the naming convention of the event handler of the widget (here the widget is TextPushButton) is as followings:

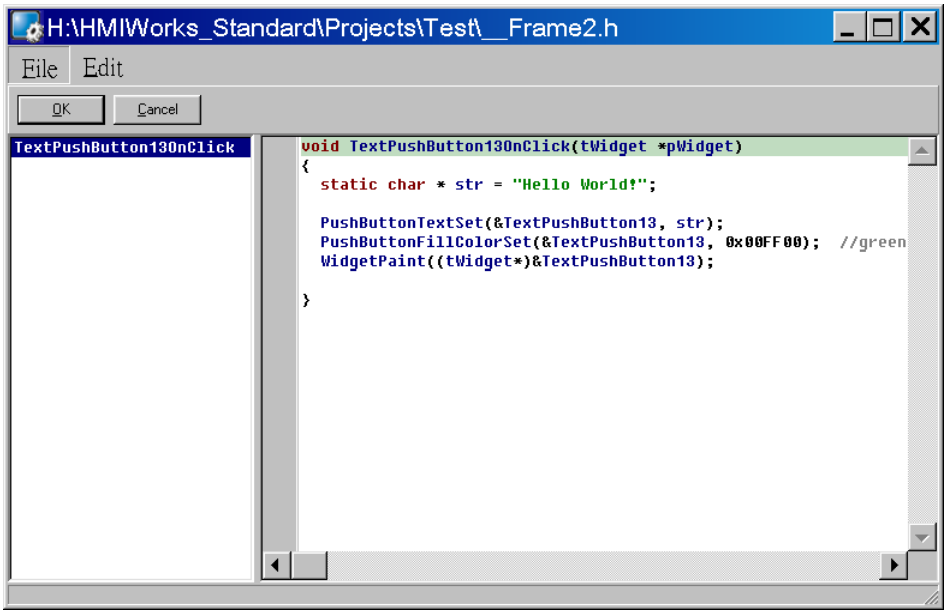


7.2.1. FillColor and Text of a TextPushButton

This section shows how to change FillColor and Text properties of TextPushButton. Simply follow the steps below.

1. Drag and drop a TextPushButton on the frame panel.

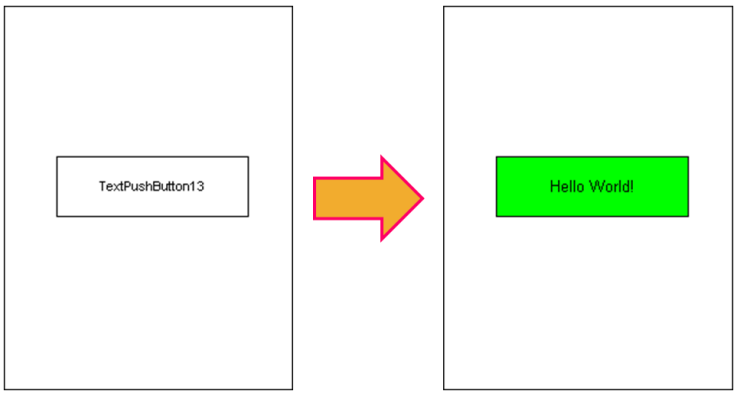
2. Double click on the TextPushButton to implement its OnClick event handler. Save the file and press OK.



3. To make clear, we copy the codes below.

```
void TextPushButton13OnClick(tWidget *pWidget)
{
        static char * str = "Hello World!";

        PushButtonTextSet(&TextPushButton13, str);
        PushButtonFillColorSet(&TextPushButton13, 0x00FF00);
        WidgetPaint((tWidget*)&TextPushButton13);
}
```

The effect of the OnClick function:



For more API functions of TextPushButton, refer to **pushbutton.h** in the following path:

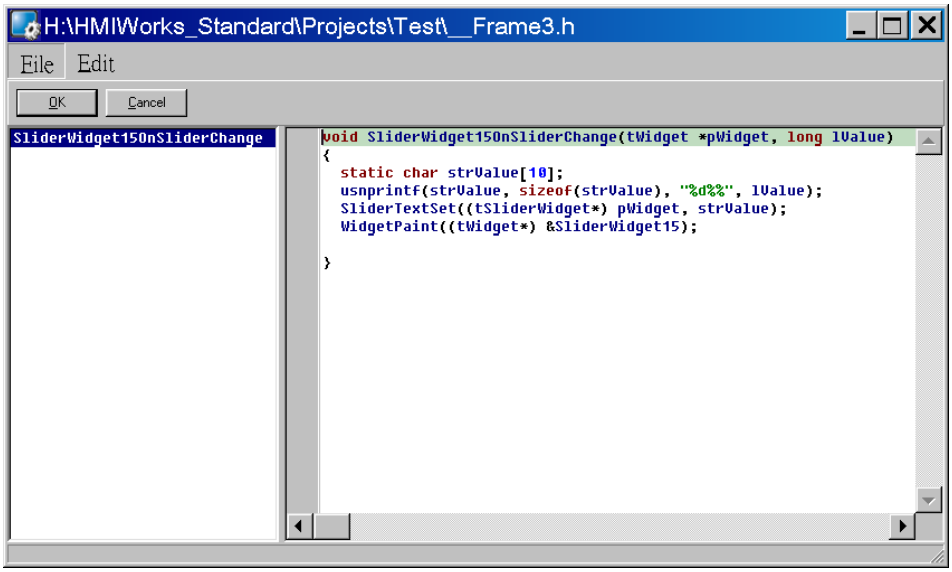
“C:\ICPDAS\HMIWorks_Standard\include\grlib”, where
“C:\ICPDAS\HMIWorks_Standard” is the installation path.

7.2.2. Percentage of a Slider

Simply follow the steps to make the percentage of the Slider changing with its position.

1. Drag and drop a Slider on the frame panel.

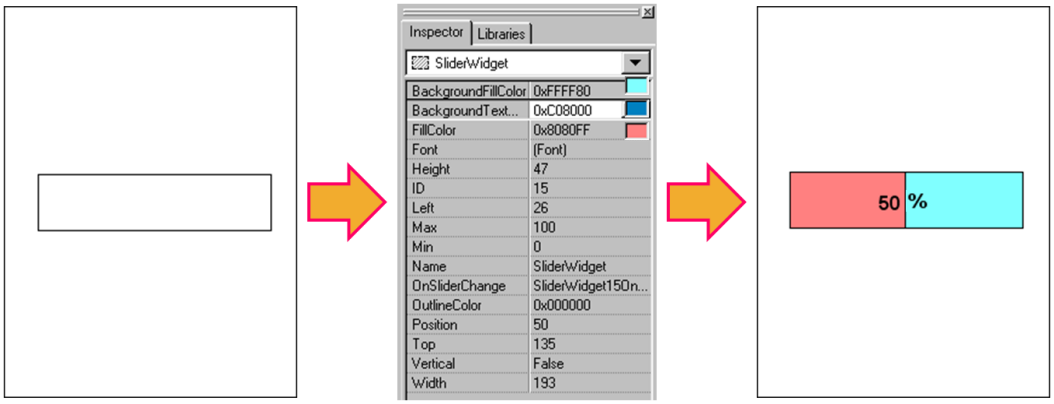
2. Double click on the Slider to implement its OnSliderChange event handler. Save the file and press OK.



3. To make clear, we copy the codes below.

```
void SliderWidget6OnSliderChange(tWidget *pWidget, long lValue)
{
   static char strValue[10];
   usnprintf(strValue, sizeof(strValue), "%d%%", lValue);
   SliderTextSet((tSliderWidget*) pWidget, strValue);
   WidgetPaint((tWidget*) &SliderWidget15);
}
```

The effect of the OnSliderChange function (after selecting colors):



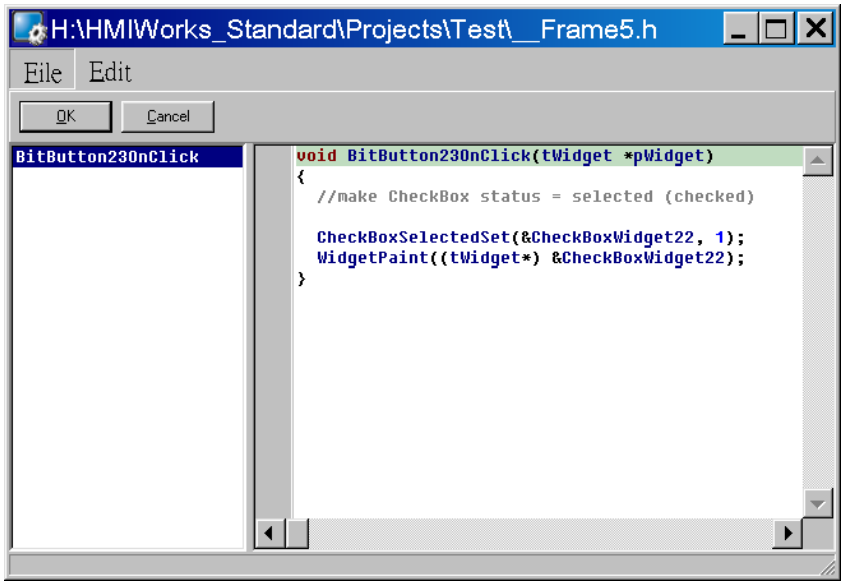
For more API functions of Slider, refer to **slider.h** in the following path: "C:\ICPDAS\HMIWorks_Standard\include\grlib", where "C:\ICPDAS\HMIWorks_Standard" is the installation path.

7.2.3. Selected of a CheckBox

Take the following steps for example to change the Selected property of the CheckBox in the run time.

1. Drag and drop a CheckBox and a BitButton on the frame panel.

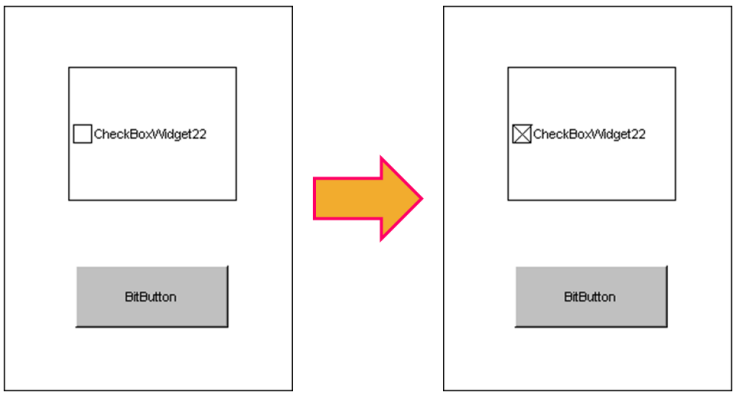
2. Double click on the BitButton to implement the OnClick event handler of the BitButton. Save the file and press OK.



3. To make clear, we copy the codes below.

```
void BitButton23OnClick(tWidget *pWidget)
{
   //make CheckBox status = selected (checked)

   CheckBoxSelectedSet(&CheckBoxWidget22, 1);
   WidgetPaint((tWidget*) &CheckBoxWidget22);
}
```

The effect of the OnClick function:

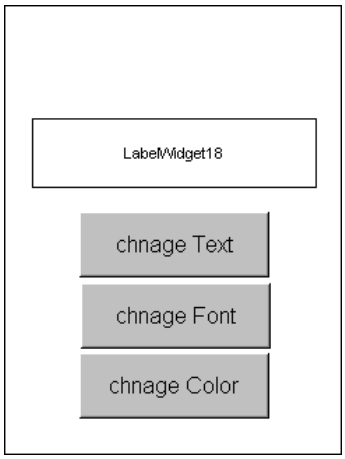


For more API functions of CheckBox, refer to **checkbox.h** in the following path:
“C:\ICPDAS\HMIWorks_Standard\include\grlib”, where
“C:\ICPDAS\HMIWorks_Standard” is the installation path.

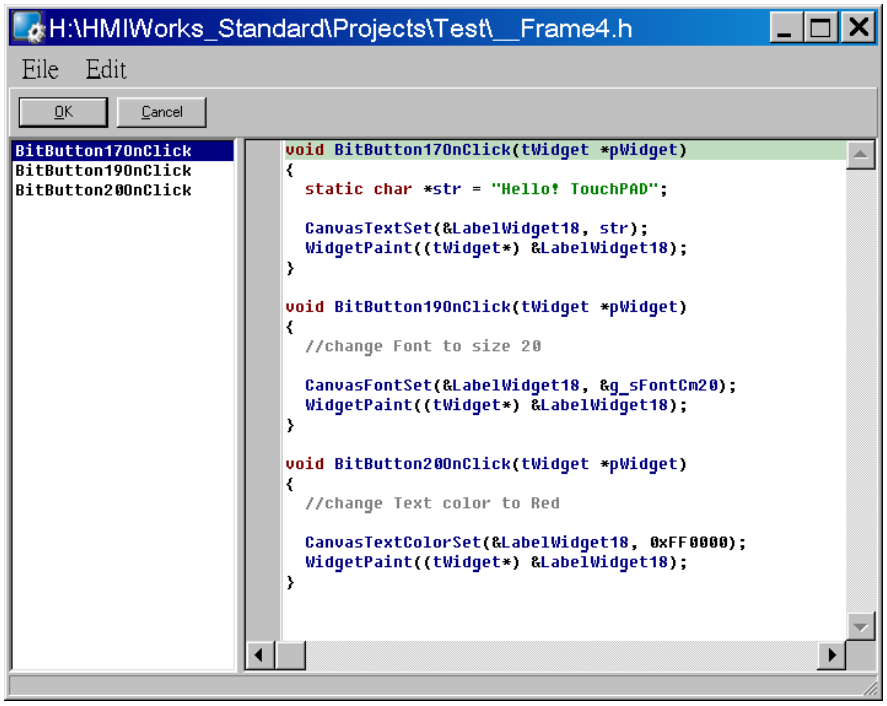
7.2.4. Font, Text and TextColor of a Label

Take the following steps for example to update properties of the Label in the run time.

1. Drag and drop a Label and three BitButton on the frame panel as below.

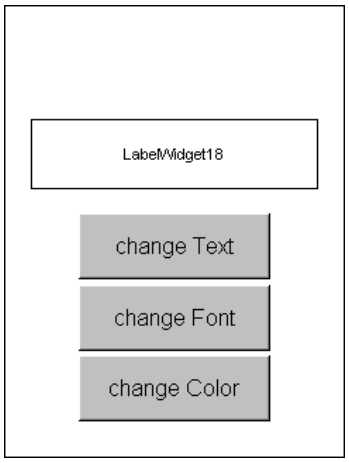
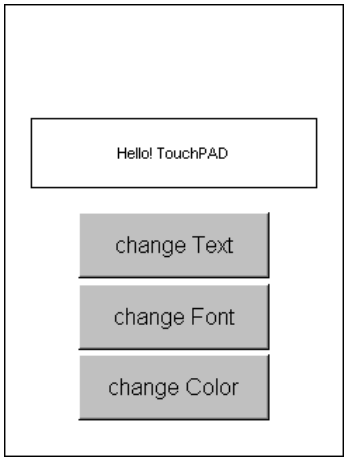


2. Double click on the BitButton to implement the OnClick event handlers of three BitButtons. Save the file and press OK.

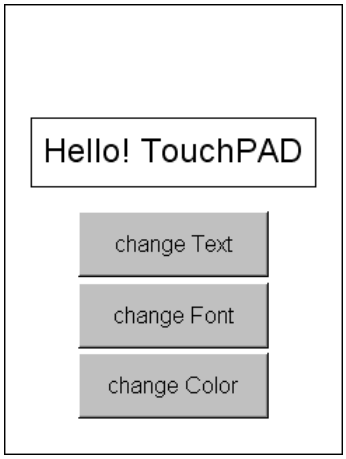


3. To make clear, we copy the codes below with their corresponding results.

Results	Codes of the event handler

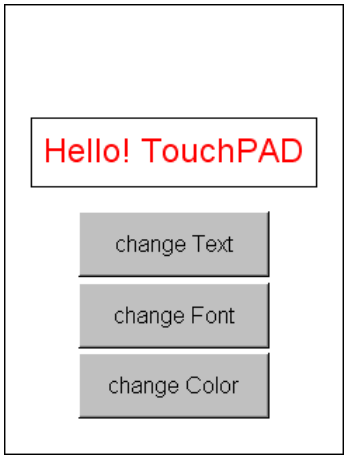
<table>
<tr>
<td> 
</td>
<td>//step 0
//the beginning snapshot
N/A</td>
</tr>
<tr>
<td> 
</td>
<td>//step 1
//Click on BitButton “change Text”

void BitButton17OnClick(tWidget *pWidget)
{
  static char *str = "Hello! TouchPAD";

  CanvasTextSet(&LabelWidget18, str);
  WidgetPaint((tWidget*) &LabelWidget18);
// or use LabelTextSet to replace
// CanvasTextSet and WidgetPaint, that is:
// LabelTextSet(&LabelWidget18, str);
}</td>
</tr>
<tr>
<td> 
</td>
<td>//step 2
//Click on BitButton “change Font”

void BitButton19OnClick(tWidget *pWidget)
{
  //change Font to size 20

  CanvasFontSet(&LabelWidget18,
&g_sFontCm20);
  WidgetPaint((tWidget*) &LabelWidget18);
}</td>
</tr>
</table>




```
//step 3
//Click on BitButton “change Color”

void BitButton20OnClick(tWidget *pWidget)
{
   //change Text color to Red

   CanvasTextColorSet(&LabelWidget18,
0xFF0000);
   WidgetPaint((tWidget*) &LabelWidget18);
}
```

For more API functions of Label, refer to **canvas.h** in the following path: “C:\ICPDAS\HMIWorks_Standard\include\grlib”, where “C:\ICPDAS\HMIWorks_Standard” is the installation path.

In the same path, there is a header file, grlib.h.
grlib.h contains prototypes for the pre-defined fonts, such as g_sFontCm20.

Appendix

Appendix is listed below:

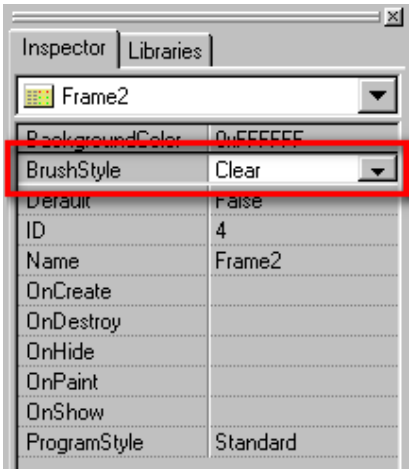
1. FAQ, Frequently Asked Questions

A.FAQ

A.1. What to do if screen flashes?

Set the “BrushStyle” property of frames to “Clear”.

BrushStyle	Solid or Clear. If BrushStyle is set to Solid, BackgroundColor is effective. However this may make screen flash if background color is quite different from the loaded picture. Setting BrushStyle Clear disables the property BackgroundColor and prevents from flashing.

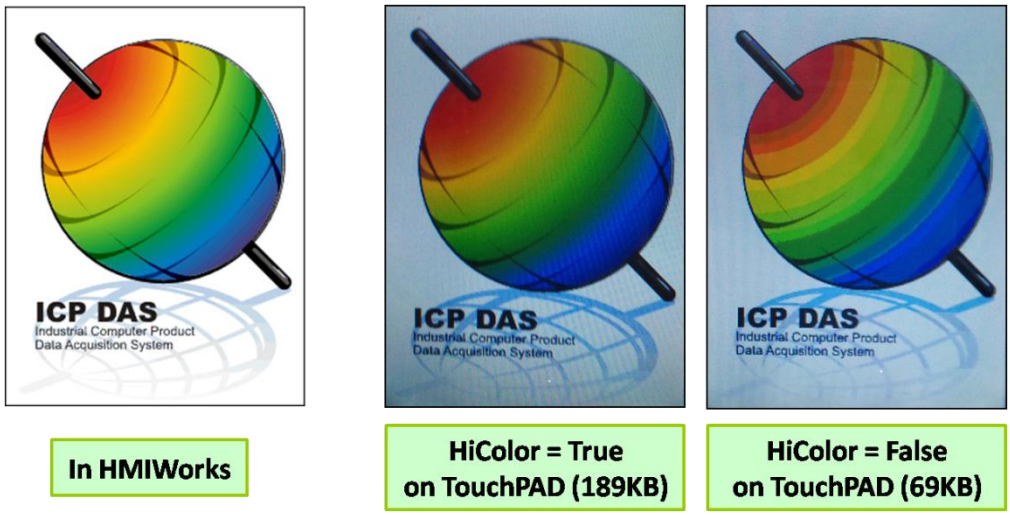


A.2. How to have higher-resolution Picture?

Set the “HiColor” property of Picture to “True”.

HiColor	True or False. This property decides whether the loaded picture is stored as 16-bit color (True) or 8-bit color (False). Default is 8-bit color.

Trade-off between firmware size and resolution



Above is the comparison between HiColor = True and HiColor = False.
The left picture is original one in HMIWorks. The two right-side pictures are real photos.
One is HiColor = True and the other HiColor = False.

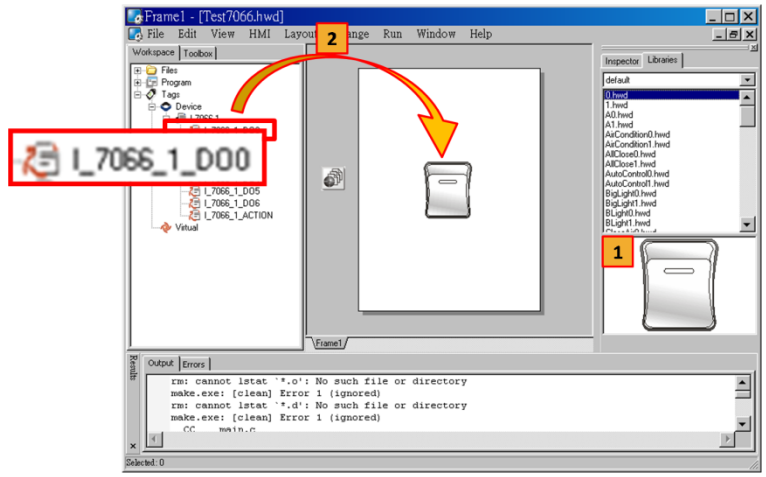
As you can see, setting HiColor to False makes the photo fragmentized while setting HiColor to True does not. Because 8-bit color does not have enough color (256 only) to represent the picture, similar colors are represented by the same color and this results in fragmentation.

However, preventing pictures from fragmentation costs TouchPAD bigger size of memory. Take above picture for example, setting HiColor to True uses memory of 189KB but setting HiColor to False costs only 69KB.

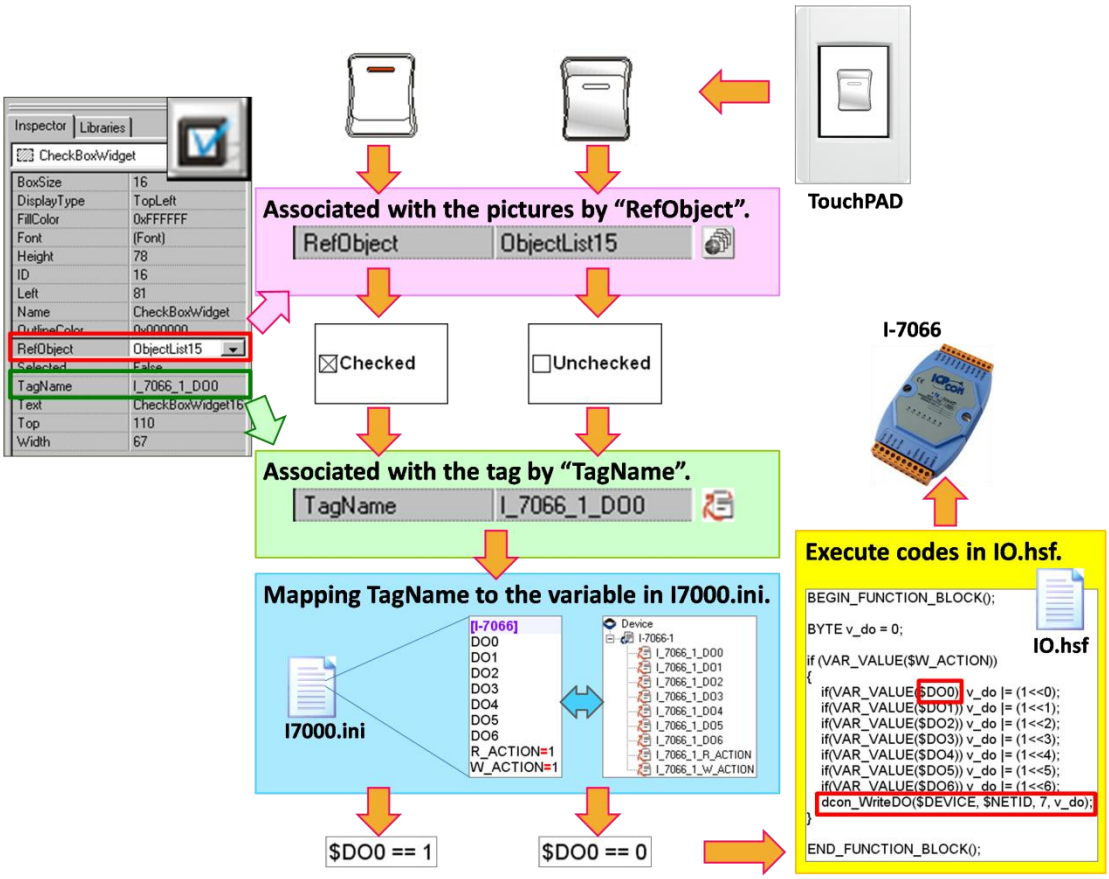
A.3. How does a TouchPAD control I/O?

From a touch to I/O:

Take I-7066 for example, **Register Devices (F3)** to automatically generate tags and then drag and drop the tag on the frame.



HMIWorks does the followings to build the relationships between TouchPAD and I/O modules.



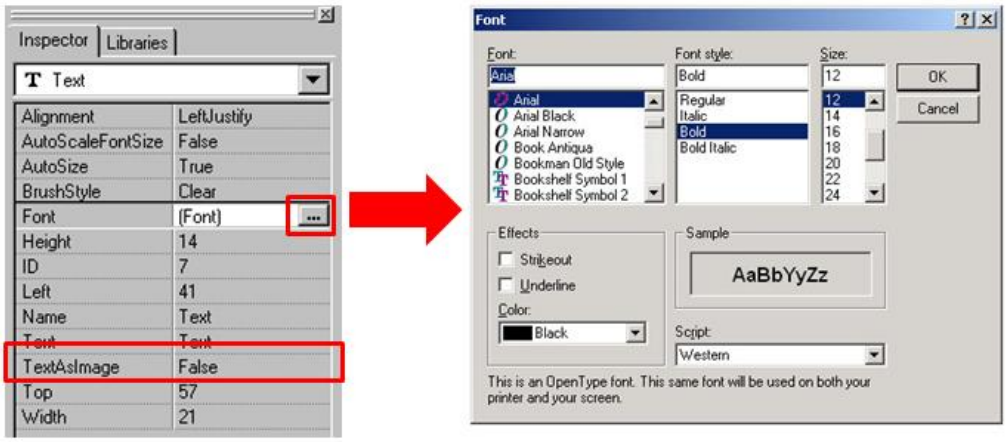
Note: the TagName property takes effect only in the programming type, Ladder. (It's easier in programming type "Standard C". Control the I/O by using API function, dcon_WriteDO, in the event handler of the CheckBox.)

A.4. How to change Font of Text?

Be sure to set TextAsImage = True. Otherwise changing font does not take effect.

Note that because of not being able to set their Text as image, the Font of **TextPushButton** and **Label** cannot change their "Font" and "Font Style" but "Size" can.

properties	description
AutoScaleFontSize	Automatically scale the font size to fit the rectangle which encloses the Text. Note: this property is enabled only when AutoSize=True.
AutoSize	True or False. This property is used to indicate that whether the size of the rectangle which encloses Text can be automatically changed to cover the whole string.
Font	The font of the Text. Note: this property is enabled only when TextAsImage=True.
TextAsImage	True or False. Whether text is stored as an image or not. If the text is treated as an image, of course, it takes more space to store and more time to download.



A.5. How to represent decimals for Ladder Designer?

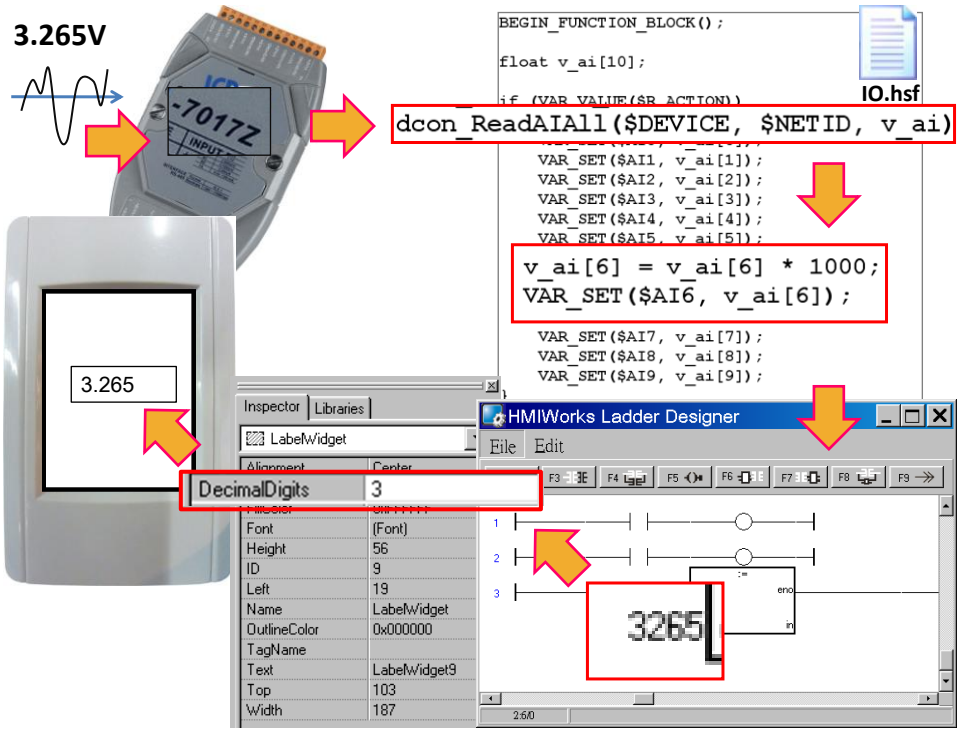
The numbers in **Ladder Desinger** are all integers. The decimals are not accepted in **Ladder Designer**. However, in some cases, users may need to calculate or display decimals. Here we provide a work-around method to handle decimals.

Take I-7017Z for example. Supposed that we use I-7017Z to read an analog value 3.265V back from a remote equipment and we want to utilize the powerful feature of HMIWorks, **Ladder Designer**. However, **Ladder Designer** supports only integers. So we must handle this drawback to directly read back the AI value from I-7017Z in **Ladder Designer**.

1. Set the property “DecimalDigits” to the number of digits in the right of the decimal point. For example, we set DecimalDigits = 3.
2. Modify the I/O module’s IO.hsf. Let the read back AI value multiplied by ten of the n-th power where n is the value of “DecimalDigits”. You can find out I/O module’s IO.hsf file in the following locations: “[HMIWorks_install_path]\ bin\Modules\”.
   For example, IO.hsf of I-7017Z is located in “C:\ICPDAS\HMIWorks_Standard\bin\Modules\I-7000\I-7017Z”, where “C:\ICPDAS\HMIWorks_Standard\” is the installation path of

HMIWorks. And we modify the IO.hsf to make v_ai[6] = v_ai[6] * 1000; Supposed we use channel 6 to read back AI value.

As the figure shows, you can see that the tag “$AI6” in **Ladder Designer** is 1000 times of the real value. With DecimalDigits set to 3, the correct value 3.265 is displayed on TouchPAD.



A.6. How to remove the startup beep of TPD-283?

TPD-283 sounds a beep when startup.
There’s no such function to disable the beep when a program starts.
Before adding this feature to the HMIWorks, we provide a workaround method to do that.

1. After finishing designing the project, press **F5** (build) instead of **F9** (run) to generate codes.
2. In the directory of the project, open the file “_frame0.c”.

3. Add one empty function which is named hmi_Beep and comment out two hmi_Beep() as the followings (then save the file.)

```
void hmi_Beep()
{
   //newly-added empty function
}

void _frame0()
{
   tContext sContext;

   //hmi_Beep();
   //hmi_Beep();

   RegisterNullWidgetMsgProc(default_timer_proc);
   …
}
```

4. Press **F10** in the HMIWorks, and cmd.exe window shows. Key in “make” to re-make the project.
5. Run → Download Only (**Ctrl** + **F9**). Then the program starts with no beep anymore.

A.7. How to store data in the flash?

Although TouchPAD does not have the feature of data logging, we do provide some segment of the flash for data saving in spite of the quite small size of the storing area.

1. For TPD-280/TPD-283:
   i. We provide users with an unsigned char array of “UserParamsData” whose length is 380 bytes. Users can store their data in it. Be sure **NOT** to exceed the length of 380 bytes or the data on the flash may be severely damaged.
   Refer to “params.h” in “C:\ICPDAS\HMIWorks_Standard\include\” for details.

ii. There's some demo code in the demo "DemoAll" which can be downloaded from ftp://ftp.icpdas.com./pub/cd/touchpad/demo/ (tpd-280.zip & tpd-283.zip). In the DemoAll project, go to the frame 4 to access the demo code. (To correctly run the project, read the readme.txt first.)

2. For TPD-280U/TPD-430/ TPD-430-EU/VPD-130/ VPD-130N:
   We have make a data storage zone of size **4KB** for users. Refer to the API reference, see these APIs: hmi_UserFlashErase, hmi_UserFlashRead and hmi_UserFlashWrite. TPD280U_UserFlash is the demo which can also be downloaded from ftp://ftp.icpdas.com./pub/cd/touchpad/demo/.