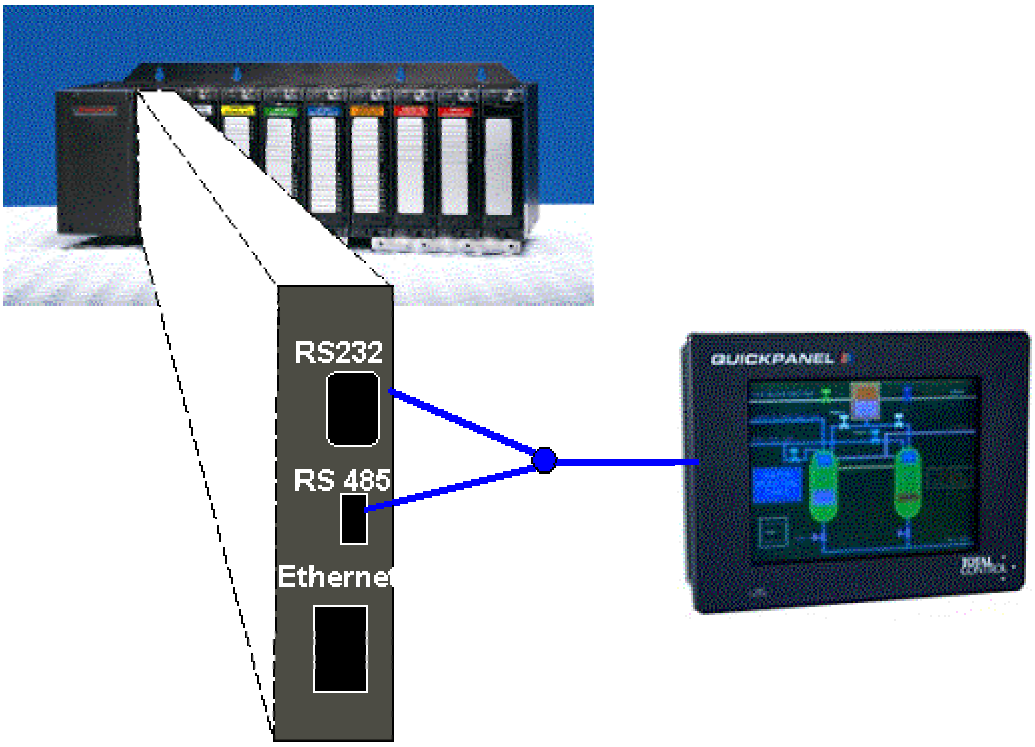
Honeywell

HC900 Hybrid Controller

When you need more than just discrete control

Product Note - HC900 with 3rd Party Interfaces



Background:
The market for operator interfaces used with PLCs and other control systems today is dynamic and aggressively competitive. These conditions have allowed users to be very selective in their choice of interface for their control system. If the Operator Interface of preference is not compatible with a new control system under evaluation, the acceptance of the new control system is less likely to occur.

Although the fully integrated operator interfaces offered by Honeywell for use with the HC900 controller offer many advantages for ease of configuration, ease of maintenance and a full range of pre-configured maintenance and diagnostic displays, they lack some of the preferred features offered by many commercially available interfaces. The most common of these include touch screen operator actions and custom graphic displays.

Problem Statements and Solutions:
There are a number of issues when using 3rd party interfaces with a controller. Some of the issues result from limitations designed into the operator interfaces while others are the result of limitations designed into the controller. Since the primary application for 3rd party interfaces has been for use with PLCs, control devices that do not mimic PLC data structures frequently encounter interfacing issues. Some of the more common issues encountered with the HC900 and 3rd party interfaces, and the attributes in Release 2.0 that offer improvements include:

1. In the initial release of the HC900, the CPU modules offered 3 communication ports, one for configuration, one for use with Honeywell operator interfaces and one Ethernet. Only the Ethernet port offered Modbus/TCP, an open protocol, while the other two ports were limited to a private protocol. This limited 3rd party interfaces to those that support Ethernet communications and Modbus/TCP protocol.

   Release 2:
   In this release, serial Modbus communication protocol is user selectable for both the RS232 configuration port and the RS485 Operator Interface port.

2. Some operator interfaces support the Modbus RTU protocol used in the HC900, but only support "holding register" data addresses up to 4096 or 32,000 (decimal). Note that the listed address range for a touch panel's Modbus RTU driver may include a leading 4 or 3 in the address range number and it may be 5 or 6 digit. In this case, the first digit really refers to the Modbus "function code". For example, as shown below, a Maple Systems panel has a range of 4096 (not 44096) for "Holding/Output Registers", so it would not cover the range needed for the HC900.

   Maple System published register range:

Controller Register Type	Controller Register Description
30001 - 34096	Input Registers, Read Only
40001 - 44096	Holding / Output Registers

   The HC900 Modbus address map has a number of parameters at the high end of the total Modbus range, including a user-defined custom register array for Signal Tags and Variables (addresses 45057-46056). To address the full database of a HC900 controller, use a 3rd party interface that supports the full Modbus range of 65535 (top of range may be listed as 65535 or 465535).

   Release 2:
   This requirement has not been impacted by changes in release 2 and remains an important selection criterion when choosing an operator interface to work with the HC900 controller.

3. In the initial release of the HC900, the majority of data provided via Modbus communications was in 4 byte floating-point format, including digital data. Many operator interfaces that were designed to work with PLCs expect to see digital data in integer format (least significant bit as a 0 or 1) for some of their graphic objects such as push-buttons, lights, panel switches, etc. These interfaces frequently could not handle the floating-point analog and digital data of the HC900.

   Release 2:
   A Modbus register array for up to 1000 data points has been added to the HC900 controller. See figure 1. Attributes of this array allow the user to specify the format of the data in each register. For digital data this includes unsigned 16 bit.

4. In addition to 3 above, some interfaces could accept floating-point analog data and display the number on a screen, but the graphic objects such as bar graphs, trend graphs, meters, etc. used on the interfaces frequently only accept integer data. This limited the amount of data that could be displayed graphically on these interfaces to those HC900 parameters that are provided in integer format, of which there were very few.

Release 2:
The Modbus Register Array discussed in number 3 also provides a solution for this problem. It allows the user to specify the analog data type format that includes: unsigned 16 bit, signed 16 bit, unsigned 32 bit, signed 32 bit and Float 32. (see figure 1)

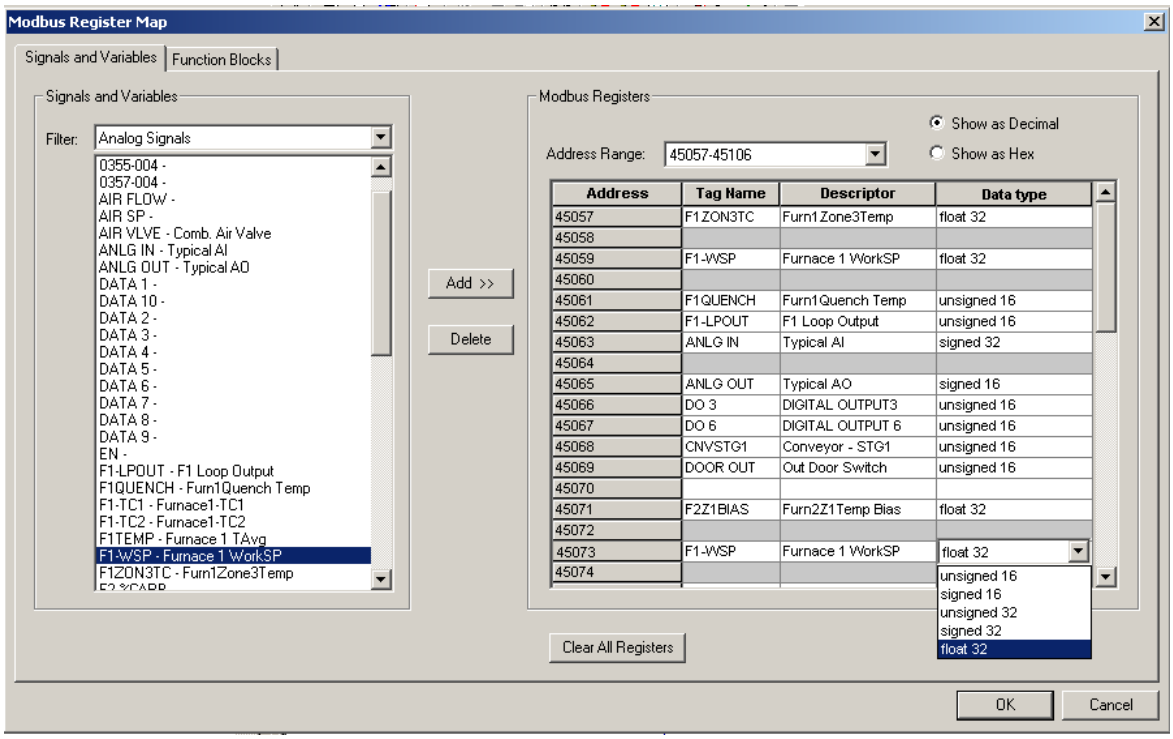


Figure 1

Notes: 1. Selections for 32 bit types are only available on Odd Registers (as shown, 45057, 45059, 45063, 45071 etc.).
2. Unused registers are permitted, allowing for future additions.(as shown,45070)
3. A single parameter may appear more than once in the table, allowing for grouping of data with the same parameter in multiple groups. (as shown 45069 and 45073)

5. As indicated in 3 above, the HC900 provided digital data in 4 byte floating-point format. Digital data is typically represented by a zero or by a one in the least significant bit. For many interfaces, they may have been able to use the floating-point format by interpreting only two bytes of the floating-point word by asking for only half of the floating-point data (one 16 bit register). The HC900 did not support this work-around. If the interface only requested 2 bytes of a floating-point data point, the HC900 always responded with a zero.

   Release 2:
   The Modbus protocol of the HC900 has been modified in release 2 to support a host device requesting only one byte of a 2 byte floating point word.

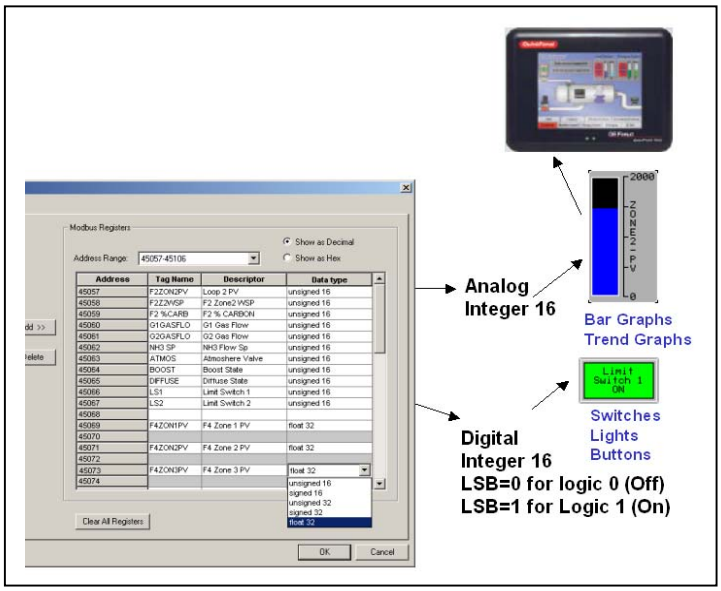


Figure 2

6. Some third party interfaces, such as those that are PC based, offer intelligent communication drivers. These interfaces could obtain performance benefits if the data in the HC900 was available in sequential Modbus addresses. The Modbus protocol supports reading multiple registers in a single operation. An example might be obtaining all of the data for a single display in a single read operation, reducing the overhead time associated with multiple reads. Since the HC900 Modbus map determines the Modbus address location and sequence for all data, this optimization feature was not available.

   Release 2:
   With the user defined Modbus Register Array the user may specify the sequence of data to take advantage of this optimization feature. This feature is particularly important to those users who plan to interface to the HC900 via modem to gather real-time data.

7. After a system has been configured and a PC application or other 3rd party interface has also been programmed, an undesirable shift in Modbus addresses would sometimes occur if the user edited his controller configuration by deleting or adding loops or other principle function blocks. This shift was caused by the way the principle function blocks are sequentially organized with the first block of a specific type being followed by the second etc. in the Modbus map. If a block is deleted, all of the block numbers above the deleted block have their Modbus block numbers reduced by one in sequence. This effectively shifts all of the Modbus register addresses for all of the data in all of the blocks above the deleted block.

   Release 2:
   In this release an additional tab has been added to the Modbus Register Map menu to select Function Blocks. From the dialog box under the Function Blocks tab, the user can position the sequence of the function blocks to re-align their address sequence to match the program of the external PC or operator interface. See figure 3.

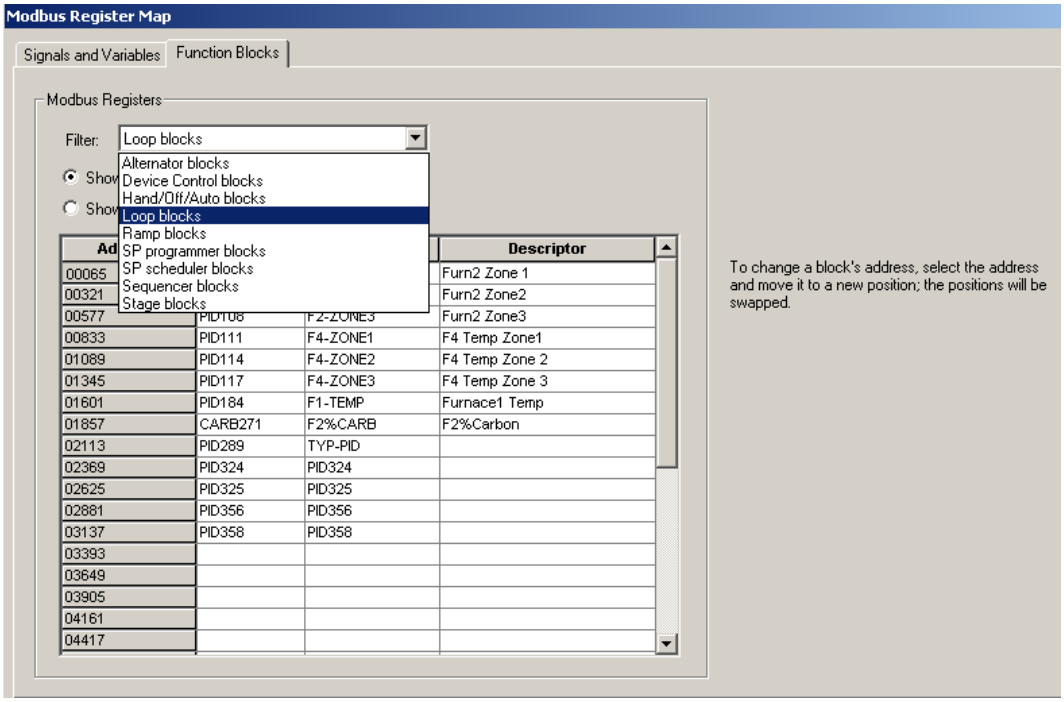


Figure 3

Setting HC900 Controller Port configuration:

1. Install HC Designer software on the PC to be used for configuration and connect the PC to the HC900 controller. Verify the ability to communicate with the HC900 controller and complete the desired control configuration including the Modbus address map.

2. Under the Utilities tab of HC Designer, select “Set Controller Serial Ports”, select a port, (RS232 or RS485), and make the desired selections:

Note: If a 559 or 1042 OI will be used with the HC900, the RS232 port must be selected.

   a. Select Modbus RTU Slave
   b. Slave type: point to point (typical)
   c. Slave address: 1 (typical)
   d. Modbus Slave double register format: (Device dependent)
   e. Port Enable = Enabled
   f. Baud Rate: = 9,600 to 57,600
   g. Stop bits = 1 or 2
   h. Parity = Odd, even, none

3. Select “SET” to write the selected parameters to the serial port. (The protocol selection for the ports is not included in the HC Designer configuration file and is not changed as part of a configuration download.)

   Note: If the controller mode switch is in the RUN or Locked positions and you are using the RS232 port for connection to the PC, once the RS232 port protocol is changed to Modbus RTU Slave, the PC will no longer be able to communicate with the controller until HC designer communications Port is changed to Modbus protocol. If the baud rate was changed when the

Modbus RTU Slave selection was made, the PC port baud rate will need to be set to this rate to reestablish communications.

3rd Party Interfacing

Many 3rd party interfaces offer specific communication drivers to support the type and model number of PLCs that they support. In addition, they commonly offer a generic Modbus communication driver for other devices. Selecting an appropriate driver is an important 1st step to interfacing to the HC900, since none of the manufactures to date that identify the HC900 as a product they support. For our validation work we used a Cutler-Hammer OI, where we selected a Modicon 984 PLC device type (this would be typical). GE, in their QuickPanel products offer a driver for the UMC800 that will support a large portion of the HC900 database, but a HC900 version was not available as of May, 2003.

Validation of the serial Modbus interface of the HC900 controller was performed with operator interfaces from the Cutler Hammer Company, Panelmate, and Automation Direct. A white paper outlining the Panelmate test experience and some hints for its use is provided below.

3rd Party OI Selection checklist:

Must – required for operation
Need – needed, but work-arounds are available if not provided.
Want – Nice to have.

1. Supports Modbus Addresses to 65535 decimal? (must)
2. Supports Modbus RTU protocol? (must)
3. Supports 4 byte floating-point data for graphic and numerical data? (need)
4. Supports two-wire RS485 communications? (need)
5. Supports RS232 communications? (need – when the configuration port is used for an OI.)
6. Ethernet communications with Modbus TCP (want)

Using Cutler-Hammer's PanelMate Operator Panels with the Honeywell HC900 Controller

PLC Name and Port Table Dialog:

Port Parameters:

1. Select the desired port number (1 or 2). This is the port number on the Op Panel that you will connect the serial cable to the HC900 controller.
2. Press the "Port Settings" button.
3. Use the following port settings:
   Electrical – pick between RS232 or 485-2 depending on the controller port to which you plan to connect the Op Panel.
   Baud Rate – 9600
   Data Bits – 8
   Stop Bits – 1
   Parity – None
4. Under "Device Use", select "Modicon RTU". Note: the Op Panel must be loaded with the Modicon RTU driver.
5. The "Local ID" field requires a Modbus station address. Enter a valid station address; just make sure it is not the same address that you will use for the HC900 controller.

PLC Parameters:

1. Press the "Add" button to add a PLC/Controller.
2. Enter a name for the controller in the "Name" field.

3. In the “Model” field select “984”. Note: the HC900 controller uses the Modbus addressing of the Modicon 984 PLC.
4. Enter the port number. This should match that which you entered in the Port Parameters. For example, if you are using port #2 on the PanelMate, enter “2” here.
5. Enter the remote ID. This is the Modbus station address associated with the HC900 controller.

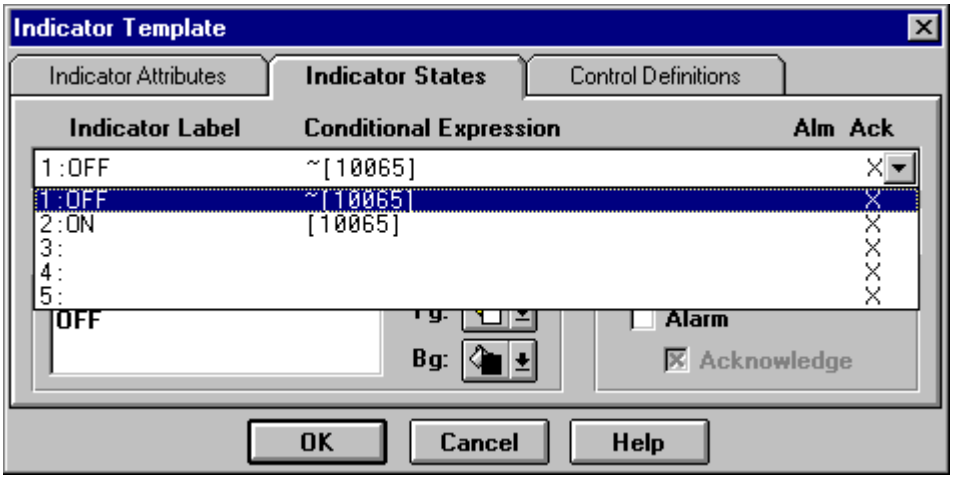
Downloading a configuration to the PanelMate:

1. After you have added or modified the Op Panel configuration you will need to save the associated screens (“Commit page changes to database?”) and the config file.
2. Select “File” and “Export”.
3. Select the desired PPS file and press “Export”. You may get a prompt about replacing the existing file.
4. Select “File” and “VCP Transfer”.
5. You’ll get a reminder about exporting the file as described in step #3 above.
6. On the PanelMate Transfer window, select the “Port Params.” Tab.
7. Select the Port Device and Baud Rate. The Port Device pertains to the PC Port used for the transfer. Note: you only need to perform this step once on a given PC.
8. On the PanelMate Transfer window, press the button “Add Configuration File to Operation List” to the right of “Download Configuration:” An operation should appear in the white area above this button. Note: the “Port” field should display the PC Port and Baud Rate to use for the transfer. The “Status” field should show “Pend”.
9. At the Op Panel (assuming it is in RUN Mode), press “Get Page”. At the numeric entry pad, press “More”. Press “Setup Page”.
10. At the Setup page on the Op Panel, press “Enter Offline Mode” and “Execute”. The unit will take a few seconds to reset.
11. At the Offline Setup page on the Op Panel, press “Enter Serial Transfer Mode” and “Execute”.
12. The Op Panel should display “***Transfer process ready”. If so, on the PC, call the PanelMate Transfer” window and press “Start”. The status should change to “Work”.
13. Wait until the status displays “Pass”. At that point, press CANCEL on the Op Panel display.
14. The Op Panel should call the Offline Mode display. Press “Enter Run Mode” and “Execute”. The Op Panel will reset to Run Mode.

Using Display Objects

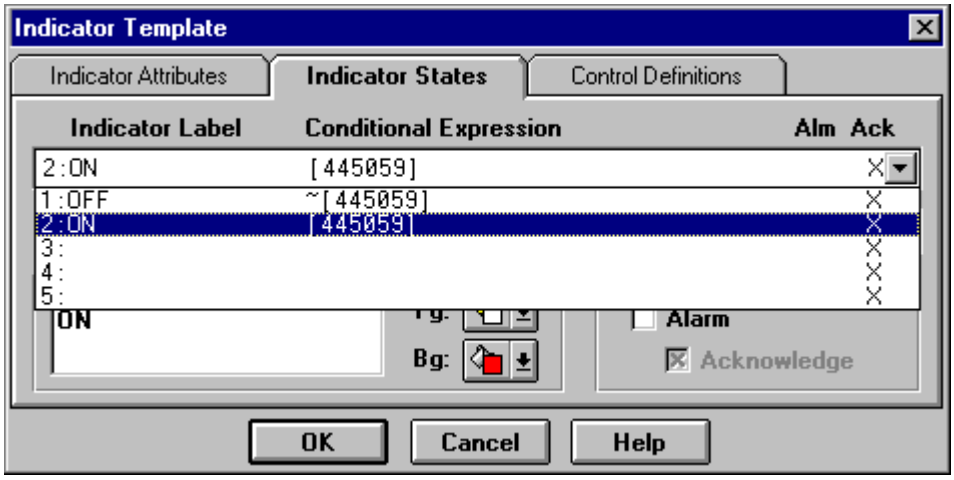
- To display a data item in floating point, enter the following in the value field: [xxxxx#fp], where xxxxx is the register address. Example [48325#fp]
- Fields with conditional entry: when you need a display object that is conditionally enterable, for example, a loop setpoint is enterable when the mode is LSP and not enterable when the mode is RSP, you can use this trick. Overlay one display object on top of another. Use the “conditional visibility” attribute on each of these two objects to “hide” the object based on another controller data item. Configure each display object with the data entry attribute as needed. For example, for Loop 1 Setpoint: use the conditional visibility expression [40253] = 1 and control type = none for one object that reads [40069#fp] as its value field, and overlay another object with conditional visibility expression [40253] = 0 and control type = numeric for one object that reads [40069#fp] as its value field. These two objects are both reading the loop mode to determine whether to display their object. Only one will display at a time. When the loop mode = LSP the field will be enterable; when the loop mode = RSP the field will be read-only.
- Do not use the Indicator template for a data item that needs to be written from the panel to the controller using a “toggle” type action. The Indicator template only supports a ONE-SHOT action. That is, if the data item is currently OFF, pressing the Indicator will write an ON (1) to the controller, but once you remove your finger from the object it will write an OFF (0) back to the controller. When you need a “toggle” action, use the “Variable Sized Control Button”. In the Control Definition tab, select “toggle” for the type attribute. For example, to toggle Loop 1 Mode between Auto and Manual, use the Variable Sized Control Button, with “toggle” type and reference [40251].

- Use the Variable Sized Control Button to call a new display (page) from the current display (page). In the Control Definition tab, select “Page Change” for the type attribute. For example, to call page 5, use the Variable Sized Control Button, with “Page Change” type and reference 5.
- To include low / high entry limits on an entry field, enter an expression in the Input Expression field on the Control Definition tab. Below is an example of setting low value = 0 and high value = 10.
  [?]*([?]>=0)*([?]<=10)+([?]<0)*[46381#fp]+([?]>10)*[46381#fp]
  The above expression allows entries from 0 to 10 when writing to register address 46381 as a float. Any entry outside of that range will be ignored. [?] indicates the entered field.
- To read from the Analog Input table: [30001#fp] will read AI #1. [30003#fp] will read AI #2.
- To read from the user-defined Modbus area: example [445061#fp]
- Readout template of an analog variable: The Value 1 field on the Expressions tab is the address of the data item to display. For example, [46341#fp]. If you need the data item to be writable from the panel, go to the Control Definition tab and enter [?] in the Input Value Expression field. In the Target Word Address field enter the Modbus address to write. Example [46341#fp]. Select a Control Type of “numeric”
- Indicator template of a DI (Input Status Area): On the Indicator States tab, you’ll need to select the drop down list and enter an expression for each state (0 and 1). Enter a conditional expression and indicator label for each of the two states. Example:



The tilda “~” character serves as a logical NOT.

- Indicator template of a digital data item in the User Modbus Area On the Indicator States tab, you’ll need to select the drop down list and enter an expression for each state (0 and 1). Enter a conditional expression and indicator label for each of the two states. Example:



- Indicator template of a DO (Coil Area): On the Indicator States tab, you'll need to select the drop down list and enter an expression for each state (0 and 1). Enter a conditional expression and indicator label for each of the two states. Example: