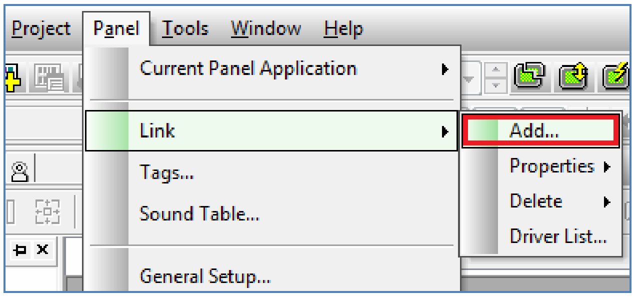
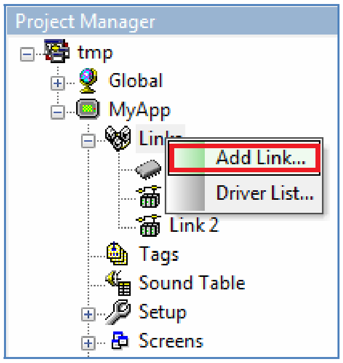
Writing Communication Programs Using Macro Commands

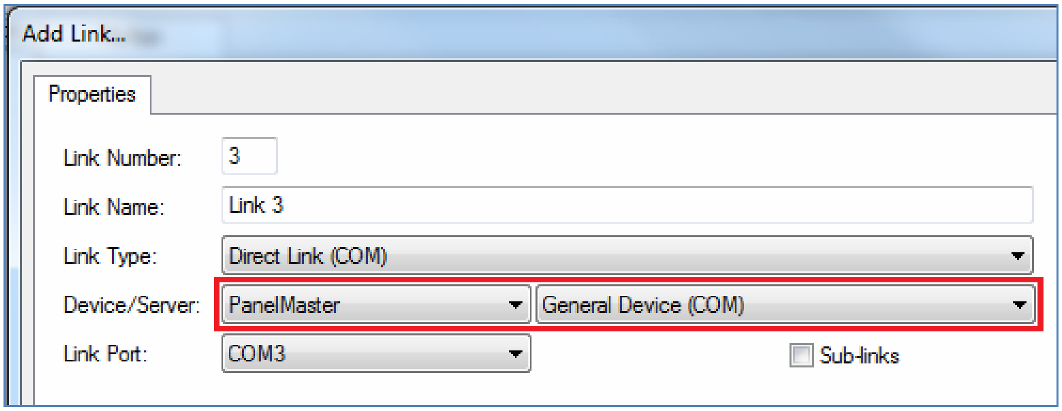
You can write a macro program to let your HMI communicate with a computing device through a serial port. For simplicity, we will use Device in the following sections to refer to the target computing device.

1. Create a communication link for the application to connect the panel and Device.



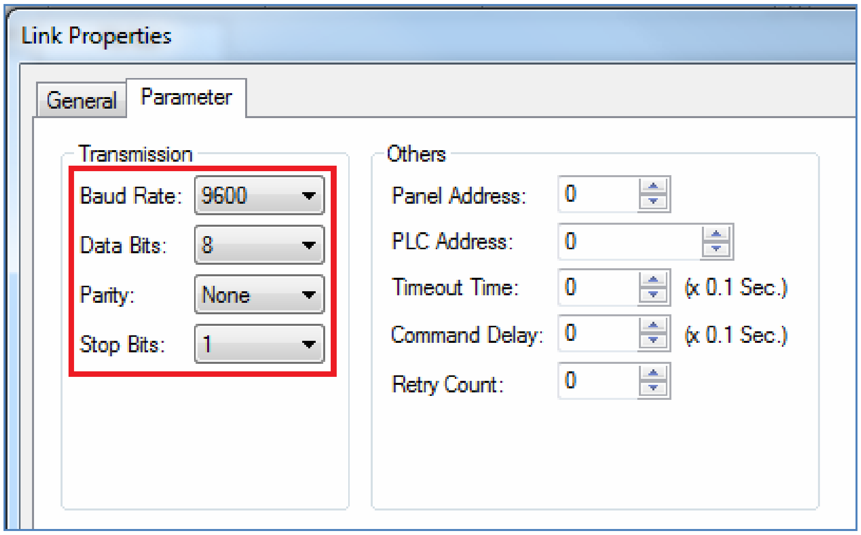


2. Specify the general settings for the link. You need to select PanelMaster General Device (COM) as the device to be connected.



Link Number: 3
Link Name: Link 3
Link Type: Direct Link (COM)
Device/Server: PanelMaster General Device (COM)
Link Port: COM3
☐ Sub-links

3. Specify the parameters of the communication.



Baud Rate: 9600
Data Bits: 8
Parity: None
Stop Bits: 1

Panel Address: 0
PLC Address: 0
Timeout Time: 0 (x 0.1 Sec.)
Command Delay: 0 (x 0.1 Sec.)
Retry Count: 0

4. Understand the key words that are available for composing the communication macro programs.

<table>
<tr><th>Key Word</th><th>Type</th><th>R/W</th><th>Function</th></tr>
<tr><td>TX</td><td>Word</td><td>Write Only</td><td>Writing a block of words to TX transmits the low bytes of that block of words to Device. The write operation is done when the transmission starts or is cancelled.

Example:

$U10 = “ABC”
$U20 = B2W($U10, 3) // Convert the byte array to a word array
TX = MOV($U20, 3) // Transmit “ABC” to Device

Operation rules:
1) The transmission cannot start until the previous transmission has finished.
2) The word TXT_STS will be set to 1 when the transmission starts.
3) When the hardware flow control is enabled (Bit EN_HS is set to 1), the transmission will not start until the device allows the panel to send data (The CTS signal on the port is asserted).
4) When the RTS/CTS flow control is enabled (Bit EN_HS is set to 1), the RTS signal on the serial port will be negated when the transmission starts and it will be asserted when the transmission ends.
5) The transmission will be cancelled if it cannot start after waiting a time specified by the word TO_TIME. The word TX_STS will be set to 2 when the timeout condition occurs.</td></tr>
<tr><td>TX_W</td><td>Word</td><td>Write Only</td><td>TX_W provides the same functionality as TX, except that the write operation is done after all the bytes are transmitted. It is recommended to use TX_W to transmit data as the macro program will not waste time to wait for the completion and the CPU of the panel will be utilized more efficiently.</td></tr>
<tr><td>TX_STS</td><td>Word</td><td>Read Only</td><td>Reading this word gets the status of the last transmission.<table><tr><th>Value</th><th>Status</th></tr><tr><td>1</td><td>Succeeded</td></tr></table></td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td><table><tr><td>2</td><td>Timeout</td></tr><tr><td>255</td><td>In progress</td></tr></table></td></tr>
<tr><td>RX</td><td>Word</td><td>Read Only</td><td>Reading this word copies a specified number of word values from the receive buffer. The receive buffer is a word array. The bytes received from Device are stored in the low bytes of the buffer. The high bytes of the buffer are always 0. Once the received data are copied, they are removed from the receive buffer. The read operation is executed whether or not there are enough bytes received in the buffer.

Example:

$U10 = MOV(RX, 3) // Receive 3 words from the receive buffer
// $U10 gets the value of the first received
// byte. $U11 gets the value of the second
// received byte. $U12 gets the value of the
// third received byte.

Operation rules:

Assume the number of words to be read from the buffer is n and the number of available words in the buffer is m.

1) If there are enough data received in the buffer, the read operation does the following 4 things: copies the first n words in the buffer to the destination, sets RX_CNT to n, sets RXB_CNT to (m-n), and sets RX_STS to 1.
2) If there are not enough data received in the buffer, the read operation does the following 4 things: copies all the available data to the destination, sets RX_CNT to m, sets RXB_CNT to 0, and sets RX_STS to 1.</td></tr>
<tr><td>RXB_CNT</td><td>Word</td><td>Read Only</td><td>Reading this word gets the number of words that are available in the receive buffer. Note that one word of the receive buffer contains one received byte.</td></tr>
<tr><td>RX_CNT</td><td>Word</td><td>Read Only</td><td>Reading this word gets the number of words that were actually taken out of the receive buffer by the last receive operation. Note that one word of the receive buffer contains one received byte.</td></tr>
<tr><td>RX_STS</td><td>Word</td><td>Read Only</td><td>Reading this word gets the status of the last receive operation.

<table><tr><th>Word Value</th><th>Description</th></tr><tr><td>1</td><td>Succeeded</td></tr><tr><td>2</td><td>Timeout</td></tr><tr><td>Others</td><td>Failed</td></tr></table></td></tr>
<tr><td>RX_W</td><td>Word</td><td>Read Only</td><td>Reading this word copies a specified number of word values from the receive buffer. The receive buffer is a word array. The bytes received from Device are stored in the low bytes of the buffer. The high bytes of the buffer are always 0. Once the received data are copied, they are removed from the receive buffer. If there are not enough data received in the buffer, the read operation will wait a time specified by TO_TIME. The read operation will not complete until all the required data are received or the timeout occurs.

Example:

$U10 = MOV(RX_W, 3) // Receive 3 words from the receive buffer.</td></tr>
</table>

<table>
<tr><td></td><td></td><td></td><td>// Wait for the data if they are not received
// from Device yet.
// $U10 gets the value of the first received
// byte. $U11 gets the value of the second
// received byte. $U12 gets the value of the
// third received byte.

Operation rules:

Assume the number of words to be read from the buffer is n and the number of available words in the buffer is m

1) If there are enough data received in the buffer, the read operation does the following 4 things: copies the first n words in the buffer to the destination, sets RX_CNT to n, sets RXB_CNT to (m-n), and sets RX_STS to 1.
2) If there are not enough data received in the buffer, the read operation will wait a time specified by TO_TIME for the required data to come. When enough data are received, the read operation performs rule 1 above. When timeout occurs, the read operation is cancelled and RX_STS is set to 2.</td></tr>
<tr><td>FLUSH</td><td>Bit</td><td>Write Only</td><td>Writing 1 to this bit flushes the receive buffer that is used to receive data from Device.</td></tr>
<tr><td>RESET</td><td>Bit</td><td>Write Only</td><td>Writing 1 to this bit resets the UART that is used for the communication between the panel and Device.</td></tr>
<tr><td>TO_TIME</td><td>Word</td><td>Read/Write</td><td>Writing the timeout time to this word sets the timeout time for the communication. The unit is 0.1 second.

Example:

TO_TIME = 20 (U) // Set the timeout time to 2 second</td></tr>
<tr><td>EN_HS</td><td>Bit</td><td>Read/Write</td><td>Writing 1 to the bit enables the RTS/CTS flow control.

<table>
<tr><th>Bit Value</th><th>Function</th></tr>
<tr><td>0</td><td>Disables the hardware flow control.</td></tr>
<tr><td>1</td><td>Enables the hardware flow control.</td></tr>
</table>

Example:

EN_HS = 1 (B) // Enable the flow control</td></tr>
<tr><td>CTS_STS</td><td>Bit</td><td>Read Only</td><td>Reading the bit gets the status of the CTS signal from the device.

<table>
<tr><th>Bit Value</th><th>CTS Signal</th><th>Comment</th></tr>
<tr><td>0</td><td>Asserted</td><td>The panel is allowed to send data to Device</td></tr>
<tr><td>1</td><td>Negated</td><td>The panel is not allowed to send data to Device</td></tr>
</table>

Example:

IF !CTS_STS (B) // Do the following when the CTS is asserted
    TX = MOV($U100, 10) // Send 10 bytes to Device
ENDIF</td></tr>
<tr><td>RTS</td><td>Bit</td><td>Read/Write</td><td>Writing 0 to the bit asserts the RTS signal on the port.</td></tr>
</table>

<table>
<tr>
<td></td>
<td></td>
<td></td>
<td>
<table>
<tr><th>Bit Value</th><th>Function</th></tr>
<tr><td>0</td><td>Asserts the RTS signal. The Device is allowed to send data to the panel.</td></tr>
<tr><td>1</td><td>Negates the RTS signal. The Device is not allowed to send data to the panel.</td></tr>
</table>
Example:
RTS = 1 (B) // Disallow Device to send data to the panel
</td>
</tr>
</table>

5. Understand the communication protocol.

6. Write the program