

Macro TRACE Function

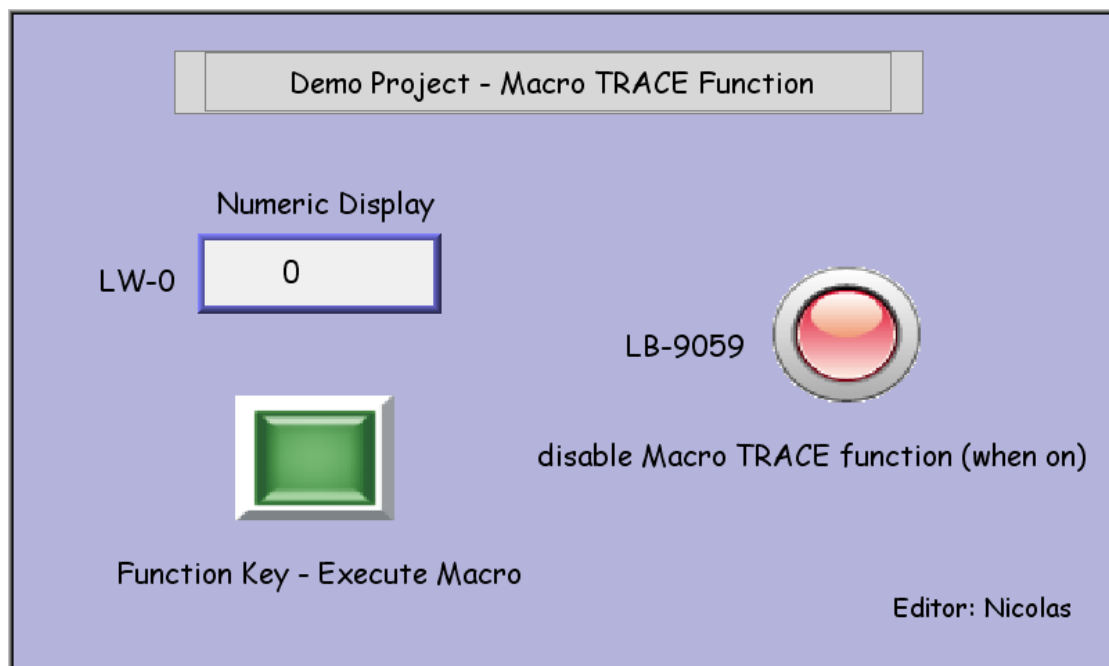
Table of Contents

1. Overview and Operation	3
2. Setting Up the Screen	8
3. Addresses	10
4. Note	11

1. Overview and Operation

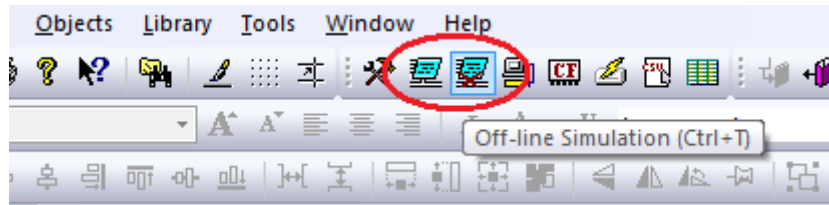
Overview

TRACE function is added to MACRO, and can be used with EasyDiagnoser, for viewing current content of the variable used.

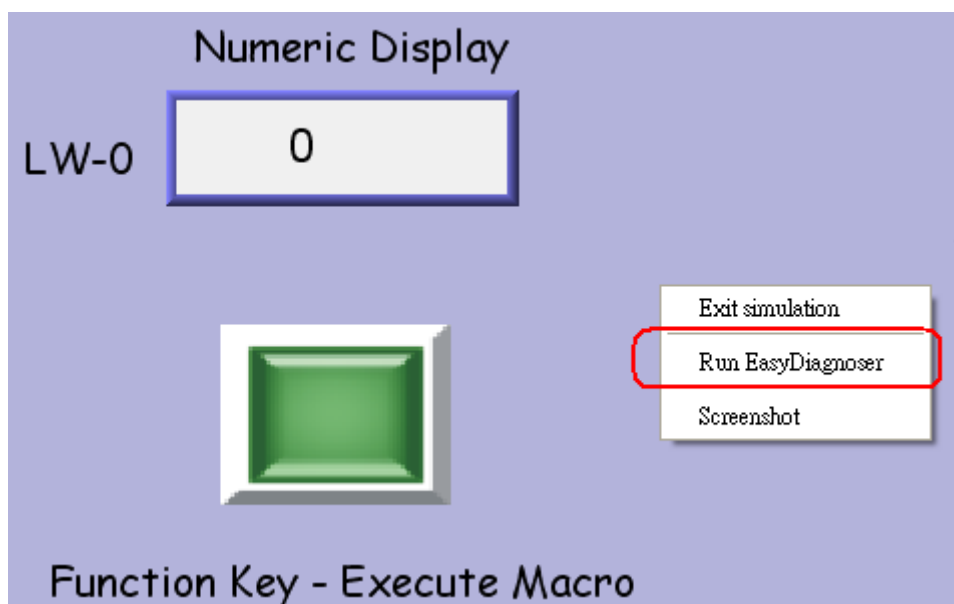


Operation

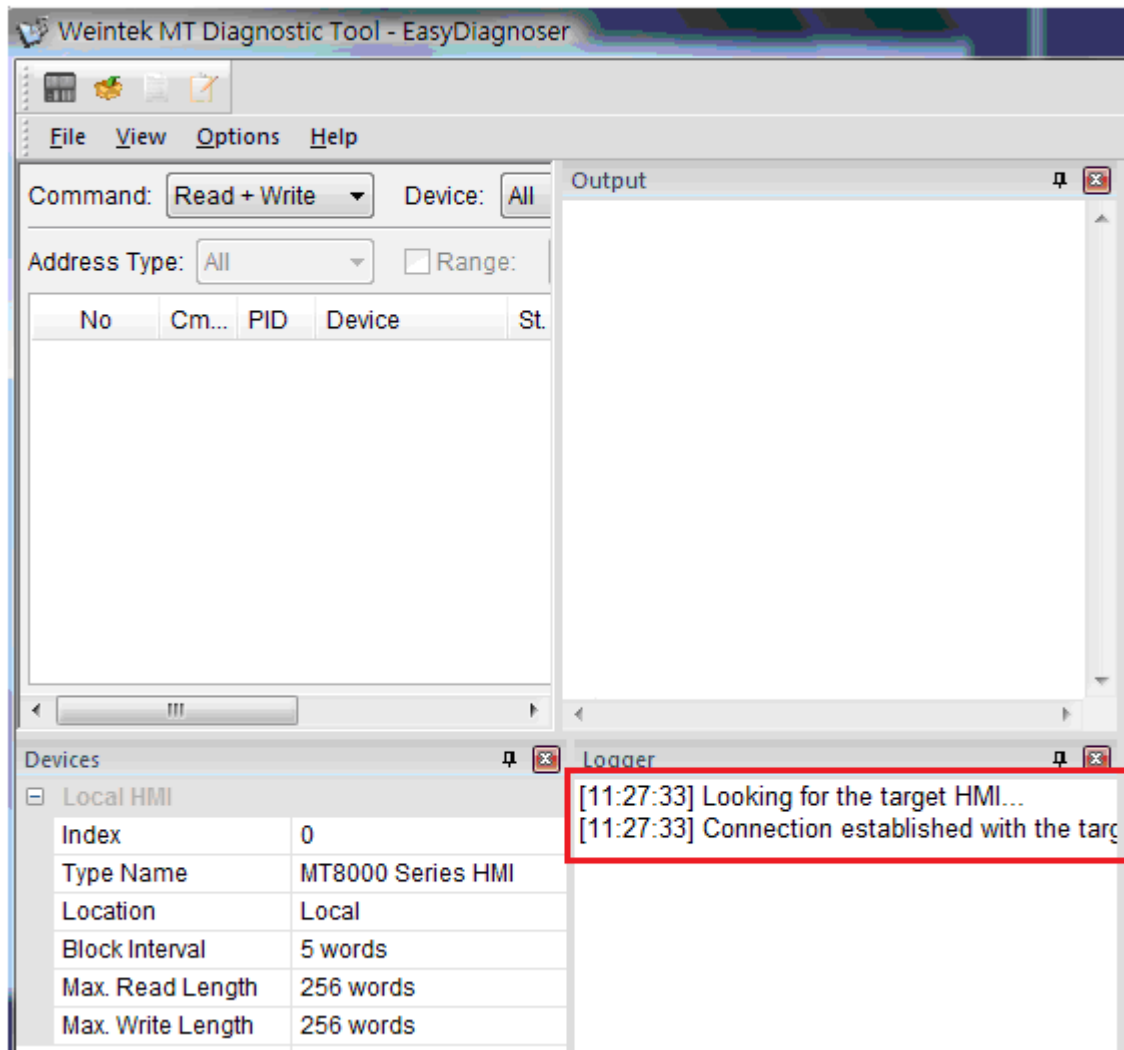
Compile the completed project and execute Off-line or On-line simulation or download to HMI.



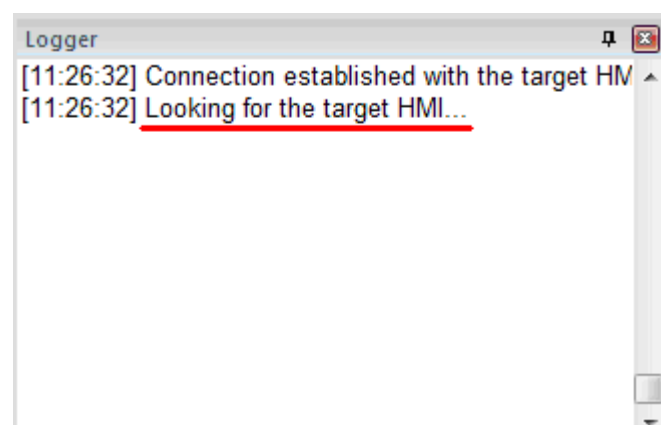
When processing simulation on PC, right click and select “Run EasyDiagnoser” in the pop-up menu.



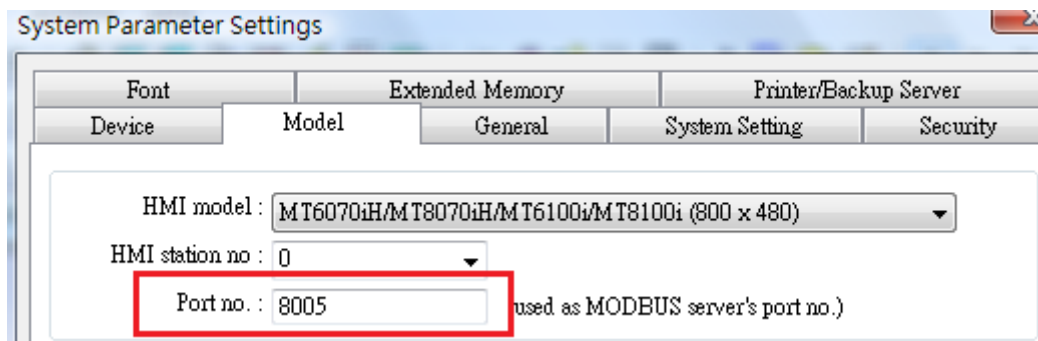
Afterwards, EasyDiagnoser will be started. [Logger] window displays whether EasyDiagnoser is able to connect with the HMI to be watched or not. [Output] window displays the output of the TRACE function. The illustration below shows that EasyDiagnoser succeeds in connecting with HMI.



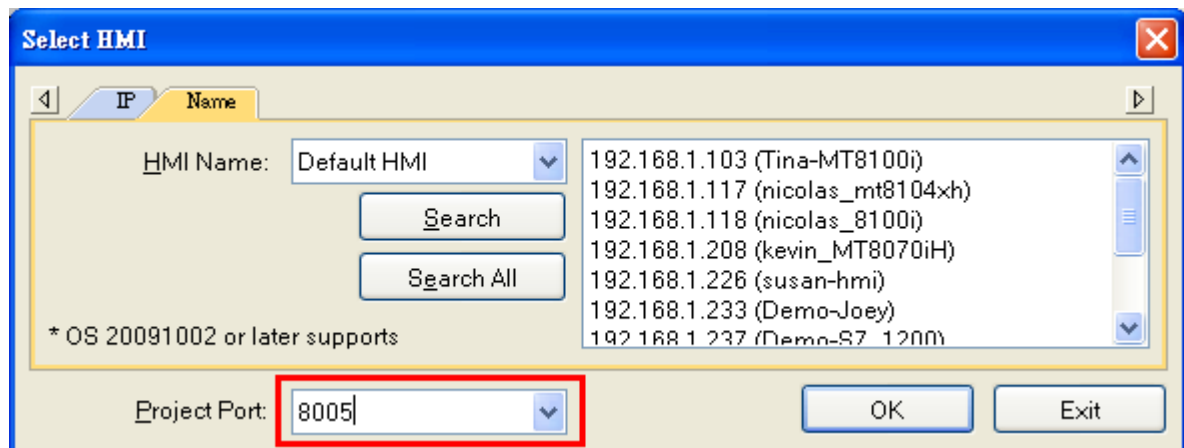
When EasyDiagnoser is not able to connect with HMI, [Logger] window displays content as shown below:



The possible reason of not being able to get connection with HMI can be failure in executing simulation on PC. Another reason is that the Port No. used in project for simulation on PC is incorrect (or occupied by system). Please change Port No. as shown, compile project then do simulation again.

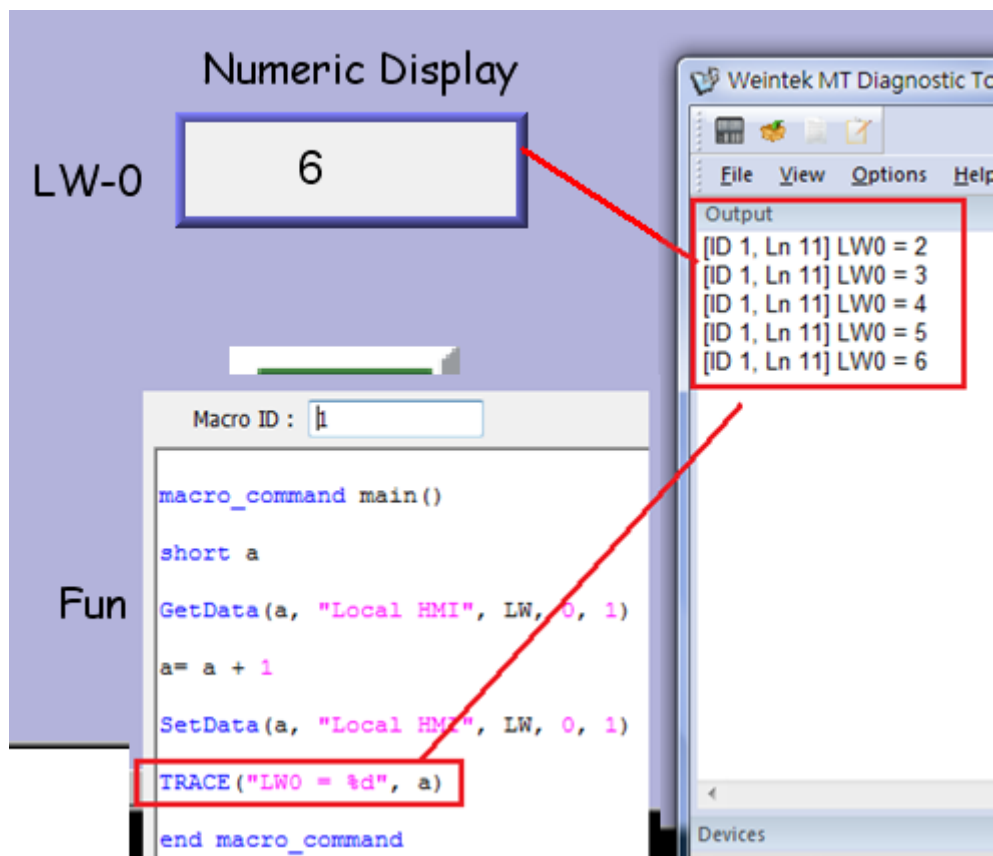


When opening EasyDiagnoser, the Port No. should be set the same as that in project. Only in this way can the communication succeed.



The three successive ports of the project port no. are preserved for HMI communication. Take the setting above as example, Port No. is set as 8005, therefore port 8005, 8006 and 8007 will be preserved. In this case when executing simulation on PC, please make sure that these ports are not occupied by other programs.

When EasyDiagnoser succeeds in connecting with HMI, simply execute macro_1, [Output] window will then display the output of the TRACE function.



2. Setting Up the Screen

The following illustrates how to use TRACE function in MACRO.

First of all, add macro_1 in the project, and in macro_1 add **TRACE** ("LW = %d", a). "%d" indicates to display current value of LW in decimal. The content of macro_1 is as the following:

```
macro_command main()

short a

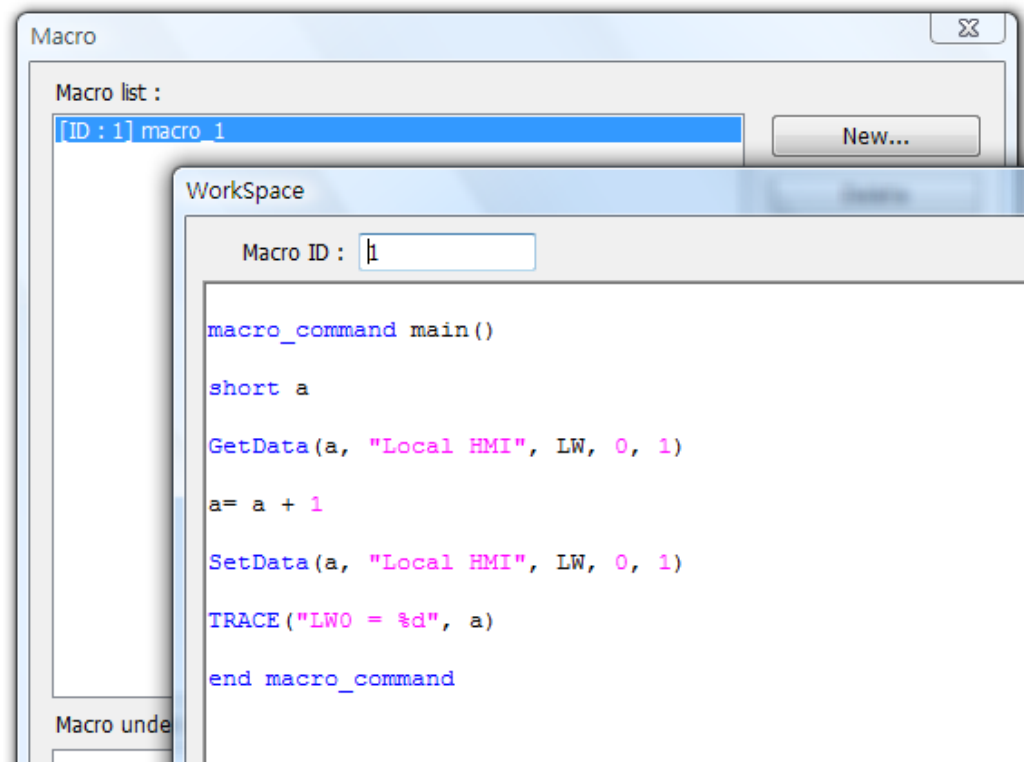
GetData(a, "Local HMI", LW, 0, 1)

a= a + 1

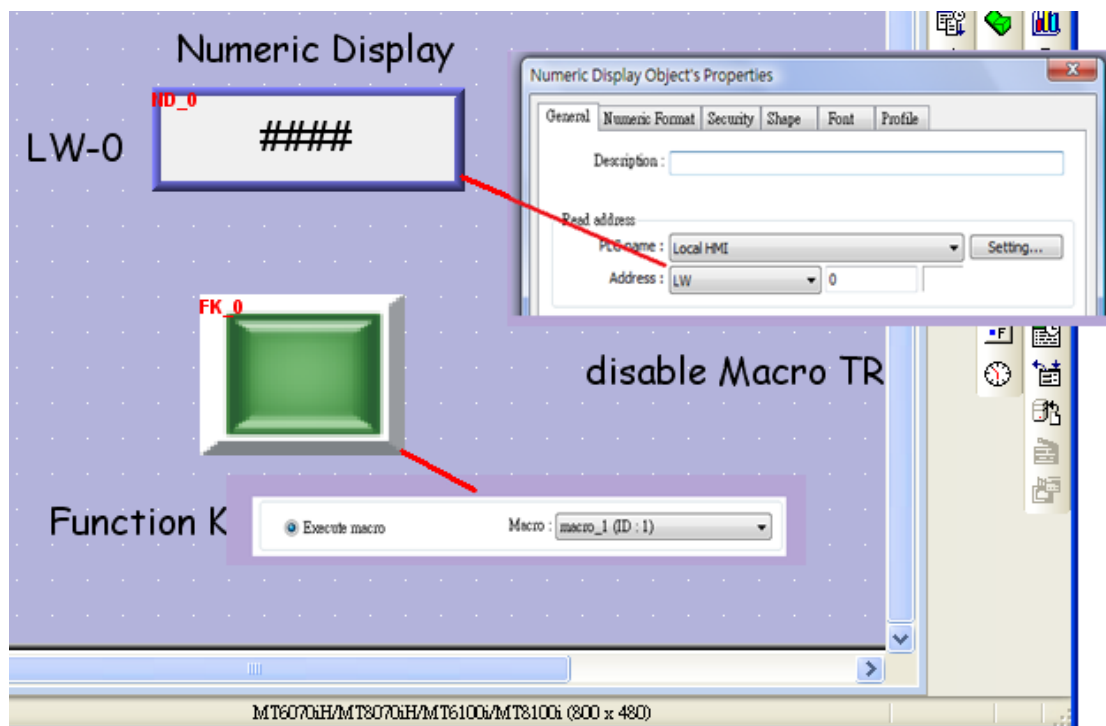
SetData(a, "Local HMI", LW, 0, 1)

TRACE ("LW0 = %d", a)

end macro_command
```



Secondly, add Numeric Display and Function Key objects in window 10 of the project. The settings of these objects are shown below. Function Key object is used to execute macro_1.



3. Addresses

The addresses of objects used in this demonstration are listed below. Users can use different addresses or object ID that suits what is needed.

Object	Address	Object ID	Description
Window 10			
Numeric display	LW-0	ND_0	Display value
Toggle Switch	LB-9059	TS_0	Enable/Disable Macro TRACE function

4. Note

Name	TRACE																
Syntax	TRACE(format, argument)																
Description	<p>Use this function to send specified string to the EasyDiagnoser. Users can print out the current value of variables during run-time of macro for debugging.</p> <p>When TRACE encounters the first format specification (if any), it converts the value of the first argument after format and outputs it accordingly. <i>format</i> refers to the format control of output string. A format specification, which consists of optional (in []) and required fields (in bold), has the following form:</p> <p style="text-align: center;">%[flags] [width] [.precision] type</p> <p>Each field of the format specification is described as below:</p> <p><i>flags</i> (optional):</p> <p style="padding-left: 40px;">- +</p> <p><i>width</i> (optional):</p> <p style="padding-left: 40px;">A nonnegative decimal integer controlling the minimum number of characters printed.</p> <p><i>precision</i> (optional):</p> <p style="padding-left: 40px;">A nonnegative decimal integer which specifies the precision and the number of characters to be printed.</p> <p><i>type</i>:</p> <table style="margin-left: 40px;"> <tr> <td>C or c</td><td>: specifies a single-byte character.</td></tr> <tr> <td>d</td><td>: signed decimal integer.</td></tr> <tr> <td>i</td><td>: signed decimal integer.</td></tr> <tr> <td>o</td><td>: unsigned octal integer.</td></tr> <tr> <td>u</td><td>: unsigned decimal integer.</td></tr> <tr> <td>X or x</td><td>: unsigned hexadecimal integer.</td></tr> <tr> <td>E or e</td><td>: Signed value having the form. [–]<i>d</i>.<i>dddd</i> e [<i>sign</i>]<i>ddd</i> where <i>d</i> is a single decimal digit, <i>dddd</i> is one or more decimal digits, <i>ddd</i> is exactly three decimal digits, and <i>sign</i> is + or –.</td></tr> <tr> <td>f</td><td>: Signed value having the form [–]<i>dddd</i>.<i>dddd</i>, where <i>dddd</i> is one or more decimal digits.</td></tr> </table>	C or c	: specifies a single-byte character.	d	: signed decimal integer.	i	: signed decimal integer.	o	: unsigned octal integer.	u	: unsigned decimal integer.	X or x	: unsigned hexadecimal integer.	E or e	: Signed value having the form. [–] <i>d</i> . <i>dddd</i> e [<i>sign</i>] <i>ddd</i> where <i>d</i> is a single decimal digit, <i>dddd</i> is one or more decimal digits, <i>ddd</i> is exactly three decimal digits, and <i>sign</i> is + or –.	f	: Signed value having the form [–] <i>dddd</i> . <i>dddd</i> , where <i>dddd</i> is one or more decimal digits.
C or c	: specifies a single-byte character.																
d	: signed decimal integer.																
i	: signed decimal integer.																
o	: unsigned octal integer.																
u	: unsigned decimal integer.																
X or x	: unsigned hexadecimal integer.																
E or e	: Signed value having the form. [–] <i>d</i> . <i>dddd</i> e [<i>sign</i>] <i>ddd</i> where <i>d</i> is a single decimal digit, <i>dddd</i> is one or more decimal digits, <i>ddd</i> is exactly three decimal digits, and <i>sign</i> is + or –.																
f	: Signed value having the form [–] <i>dddd</i> . <i>dddd</i> , where <i>dddd</i> is one or more decimal digits.																

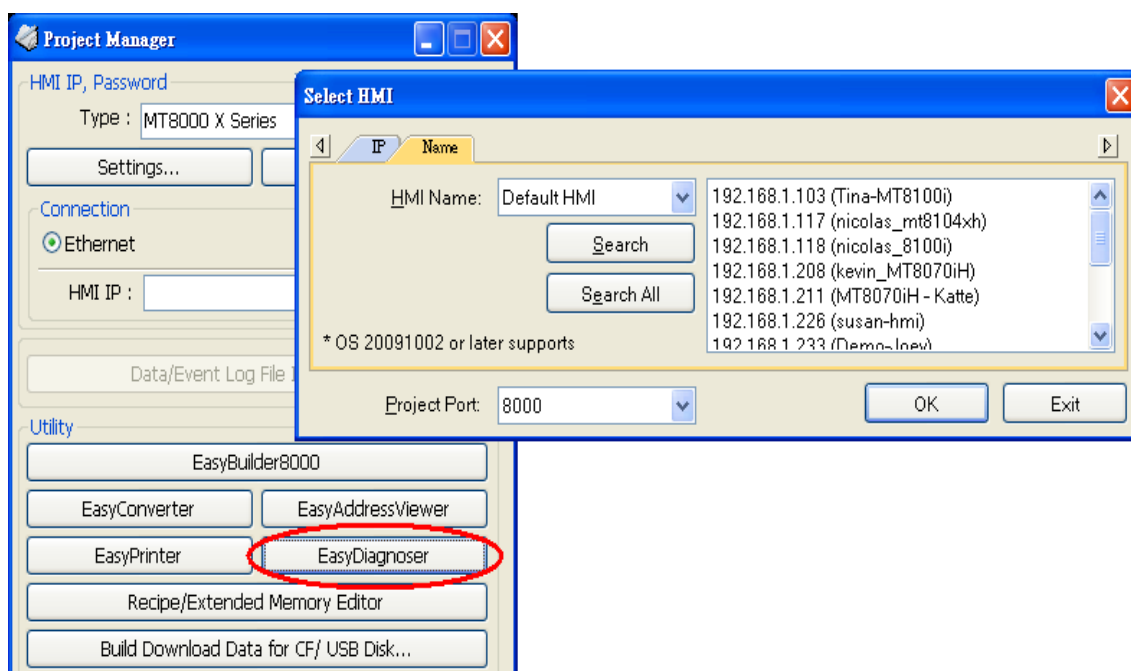
	<p>The length of output string is limited to 256 characters.</p> <p>The <i>argument</i> part is optional.</p>
Example	<pre>macro_command main() char c1 = 'a' short s1 = 32767 float f1 = 1.234567 TRACE("The results are") // output: The results are TRACE("c1 = %c, s1 = %d, f1 = %f", c1, s1, f1) // output: c1 = a, s1 = 32767, f1 = 1.234567 end macro_command</pre>

1. Newly Added LB9059 – disable MACRO TRACE function (when ON)

When set ON, the output message of TRACE won't be sent to EasyDiagnoser.

2. Users can directly execute EasyDiagnoser.exe from Project Manager. In Project Manager, current HMI on line will be listed; users can simply select the HMI to be watched.

Please note that Project Port should be the same as Port No. used in project file.



3. Download project to HMI to start operating. When EasyDiagnoser is unable to get connection with the HMI to be watched, it is possible that HMI power is not ON, or Port No. is incorrect. This may cause EasyDiagnoser to connect then disconnect with HMI continuously. Please check if the Port No. in EasyDiagnoser settings is same as that of the project. The way to change it is described before.