31.cMT Diagnoser

This chapter explains cMT Diagnoser which allows diagnostic operations in cMT / cMT X Series Model.

31.1.	Overview	31-2
31.2.	How to Start Diagnoser	31-2
31.3.	cMT Diagnoser User Interface	31-4
31.4.	Additional Notes	31-18



31.1. Overview

cMT Diagnoser can monitor data and status of all objects in the current window when it is connected to an HMI, show the status of individual communication packet, show communication status with MQTT server, and debug macros.

31.2. How to Start Diagnoser

cMT Diagnoser can be used in simulation mode or on HMI. When it is used on HMI, it can be opened from within cMT Viewer, or run an independent program named as cMT Diagnoser be directly opened on HMI.

1. In on-line/off-line simulation, cMT Diagnoser can be opened by clicking the right mouse button in cMT Viewer simulation screen and then click Diagnoser.



2. To permit connection to HMI from cMT Diagnoser, enable the Diagnoser option which is found in the [System Parameters] » [Remote] tab in EasyBuilder Pro.



Extended Me	mory Cel	llular Data Network	Time Sync./DST	e-Mail	Recipes
Device	Model	General	System Setting	Remote	Security
Prohibit	remote HMI con password remote password remote	necting to this machi e-read operation (or e-write operation (or	ne set LB9053 on) set LB9054 on)		
VNC server		8			
Pacowor	d free				
	d from project				
	modo				
Wonton	mode				
EasyAccess s	erver				
		Location of	f EasyAccess 2.0 server :	Global	•]
Diagnoser					
👽 Enable					
Passwor	d free				
			Pass	word : 111111	
				07	

After downloading the project to HMI, cMT Viewer can open cMT Diagnoser. If [Password free] is not selected in Remote tab in previous step, a password is required for running cMT Diagnoser.



3. Execute cMTDiagnoser.exe, which can be run independently, in EasyBuilder Pro installation directory, and find an HMI to connect to.





31.3. cMT Diagnoser User Interface

31.3.1. Object Settings

1. In the Object tab in cMT Diagnoser user interface, the objects used in the current window and their addresses, properties, and statuses are shown in a table.

💶 cMT Diagnoser					- • •
Cobject Device Packet	Macro N	AQTI			203
Window 10: WINDOW_010	•				+ 🖬
Name	Address	Туре	Length	Value	
 Global Objects Window 4: Common Window Window 10: WINDOW_010 Numeric (3) Local HMI 	LW-0	16-bit Unsigned	1	300	
 Option List (5) Local HMI Numeric (6) 	LW-10	16-bit Unsigned	1	0	
Local HMI Numeric (8)	LW-20	16-bit Unsigned	1	0	
Local HMI Toggle Switch (9)	LW-10	16-bit Unsigned	1	0 false	
▲ Numeric (10)			1	alse	
Numeric (11)	LVV-30	To-bit Signed	1	U	
Local HMI A Numeric (12)	LW-40	16-bit Unsigned	1	0	
Local HMI A Numeric (13)	LW-50	16-bit Signed	1	0	
Local HMI Numeric (14)	LW-60	16-bit Unsigned	1	0	
Local HMI Numeric (15)	LW-70	32-bit Signed	1	0	
Local HMI	LW-80	32-bit Unsigned	1	0	

2. Selecting a window from the drop-down list can quickly change to another window.



. 💶 (Diagnoser		
<	Object	Device	Packet
	Window 10: WI	INDOW_010	•
	🛅 Window 10: WI	NDOW_010	
	🛅 Window 50: Key	/pad 1 - Integer	
	🛅 Window 51: Key	/pad 2 - Integer	
	🛅 Window 52: Key	/pad 3 - Integer	
	🛅 Window 53: Key	/pad 4 - Integer	
	🛅 Window 54: Key	/pad 5 - Integer	
	🛅 Window 55: Key	/pad 6 - Integer	
	🛅 Window 56: Key	/pad 7 - HEX	
	🛅 Window 57: Key	/pad 8 - Floating	
	🛅 Window 58: Key	/pad 9 - Numeric	

3. The object selected in the cMT Diagnoser window will be highlighted as shown below.

CMT Viewer			0000 00 10 11 0				
+1	300 +10	+100	<i>2029</i> (219 11 0)				
plus 1	Diagnoser	_	_	_	_		
	Object Device	e Packet	Macro	MQTT			1000
0	Window 10: WINDOW_01	0	•			+ 🗇	
	Name	Address	Туре	Length	Value		
	 Global Objects Window 4: Commo, Window 5: Device Window 10: WIND Numeric (3) 						
	Local HMI	LW-0	16-bit Uns	1	300		
	Local HMI	LW-10	16-bit Uns	1	0		
data type	Local HMI	LW-20	16-bit Uns	1	0		
	Local HMI	LW-10	16-bit Uns	1	0		
6006077	Local HMI	LB-0	Bit	1	false		
	Local HMI	LW-30	16-bit Sig	1	0		
	Local HMI Numeric (12)	LW-40	16-bit Uns	1	0		

4. The value of an object can be changed by double-clicking on it in cMT Diagnoser window.



300 +1 +10 +1 +10 pust	Civit viewer			2020-02-19 11.0	7:24			
Image: Control of the second		+1 +10	+100					
Object Device Packet Macro MQTT Image: Comparison of the state of the sta		Diagnoser						
Image: Second		Object Device						
Name Address Type Length Value Global Objects Window 4: Commo Window 5: Device Vindow 10: WIND • Window 10: WIND • Window 10: WIND • Window 10: WIND • Window 10: WIND • Numeric (3) Local HMI LW-0 16-bit Uns 1 0 Local HMI LW-20 16-bit Uns 1 0 1000 1000 Local HMI LW-20 16-bit Uns 1 0 1000 1000 1000 Local HMI LW-20 16-bit Uns 1 0 1000		0 Window 10: WINDOW_01	0				+ 0	
Global Objects Window 4: Commo Window 5: Device Window 10: WIND • Window 10: WIND • Window 10: WIND • Numeric (3) Local HMI LW-0 Local HMI LW-10 • Numeric (6) Local HMI LW-20 Local HMI LW-20 • Numeric (8) Local HMI LW-10 Local HMI LW-10 • Numeric (8) Local HMI LW-20 Local HMI LW-20 • Numeric (7) Local HMI LW-10 • Numeric (10) Local HMI LW-20 • Numeric (10) Local HMI Local HMI LW-30 • Numeric (11) Local HMI Local HMI LW-30 • Numeric (11) Local HMI Local HMI LW-40 • Numeric (11) Local HMI Local HMI LW-40 • Numeric (11) Local HMI Local HMI LW-40 • Numeric (11) Local HMI <td></td> <td>Name</td> <td>Address</td> <td>Туре</td> <td>Length</td> <td>Value</td> <td></td> <td></td>		Name	Address	Туре	Length	Value		
Local HMI LW-0 16-bit Uns 1 B00 (*) Option List (5) Local HMI LW-10 16-bit Uns 1 0 Aumeric (6) Local HMI LW-20 16-bit Uns 1 0 Numeric (8) Local HMI LW-10 16-bit Uns 1 0 - Toggle Switch (9) Local HMI LB-0 Bit 1 false - Numeric (10) Local HMI LW-30 16-bit Sig 1 0 - Numeric (11) Local HMI LW-30 16-bit Sig 1 0		 Global Objects Window 4: Commo Window 5: Device Window 10: WIND Numeric (3) 						
data type Local HMI LW-10 16-bit Uns 1 0 • Numeric (6) Local HMI LW-20 16-bit Uns 1 0 • Numeric (8) Local HMI LW-10 16-bit Uns 1 0 • Toggle Switch (9) Local HMI LB-0 Bit 1 false • Local HMI LW-30 16-bit Sig 1 0 • Numeric (10) Local HMI LW-30 16-bit Sig 1 0 • Numeric (11) Local HMI LW-40 16-bit Sig 1 0		Local HMI	LW-0	16-bit Uns	1	300		
data type Local HMI LW-20 16-bit Uns 1 0 • Numeric (8) Local HMI LW-10 16-bit Uns 1 0 • Toggle Switch (9) Local HMI LB-0 Bit 1 false • Numeric (10) Local HMI LW-30 16-bit Sig 1 0 • Numeric (11) Local HMI LW-30 16-bit Sig 1 0 • Numeric (11) Local HMI LW-40 16-bit Uns 1 0		Local HMI	LW-10	16-bit Uns	1	0		
Local HMI LW-10 16-bit Uns 1 0 Toggle Switch (9) Local HMI LB-0 Bit 1 false Numeric (10) Local HMI LW-30 16-bit Sig 1 0 Numeric (11) Local HMI LW-40 16-bit Uns 1 0	data t	Local HMI	LW-20	16-bit Uns	1	0		
Local HMI LB-0 Bit 1 false Numeric (10) Local HMI LW-30 16-bit Sig 1 0 Numeric (11) Local HMI LW-40 16-bit Uns 1 0		Local HMI	LW-10	16-bit Uns	1	0		
Local HMI LW-30 16-bit Sig 1 0 • Numeric (11) Local HMI LW-40 16-bit Uns 1 0		Local HMI	LB-0	Bit	1	false		
Local HMI LW-40 16-bit Uns 1 0		Local HMI	LW-30	16-bit Sig	1	0		
Numeric (12)		Local HMI Numeric (12)	LW-40	16-bit Uns	1	0		

In cMT Diagnoser, watch addresses can be added or deleted. Click + to add a new address, or select an address and then click to delete it.

🗈 cMT Diagnoser					×
< Object Device Packet	Macro I	NQTT			ŝ
Window 10: WINDOW_010	•				+ 🖻
Name	Address	Туре	Length	Value	
 Global Objects Window 4: Common Window Window 10: WINDOW_010 Watch 					
Local HMI	LB-0	Bit	1	false	





31.3.2. Device Settings

1. The state and property of the device communicating with HMI is shown in the Device tab.

cMT Diagnoser		
Object Device Pack	et Macro MQTT	
Property	Value	
riopeny	Value	
 Local HMI 		
Index	0	
State	Connected	
Location	Local	
Device Type	cMT3090	
Interface		
Block Interval	256	
Max. Read Length	256	
Max. Write Length	256	
DELTA DVP		
Index	1	
State	Disconnected	
Location	Local	
Device Type	DELTA DVP	
Interface	COM 1 (9600, E, 7, 1)	
Block Interval	5	
Max. Read Length	32	
Max. Write Length	32	

31.3.3. Packet Settings

In the Packet tab, select the ranges and then click
 to observe communication status between HMI and PLC.

Activity	Pollir	Ig							
Packet Typ	be: All		• Device:	All	•				
\ddress Typ	be: All		 Keyword: 						Ch-
No.	Туре	PID	Device	Station	Address	Length	Index	Time	Result
222	Р	200063	Local HMI		PLW-8950	1		0	Success
221	Р	200065	Local HMI	10	LW-0	150	6	16	Success
220	Ρ	200064	Local HMI	-	LB-0	1	-	16	Success
219	Ρ	200062	Local HMI	~	LB-8999	1		33	Success
218	Ρ	200063	Local HMI))酒	PLW-8950	1		0	Success
217	Р	200067	Local HMI	10	LB-12360	1	6	17	Success
216	Ρ	200065	Local HMI	-	LW-0	150	-	16	Success
215	Р	200064	Local HMI	æ	LB-0	1		33	Success
214	Р	200063	Local HMI	15 15	PLW-8950	1	10	0	Success
213	Р	200062	Local HMI	1	LB-8999	1	6	17	Success
212	Р	200067	Local HMI	2	LB-12360	1	-	17	Success

ltem	Description
Packet	All
Туре	Displays all packets in Activity area.
	Read



	Displays only Read packets in Activity area.
	Write
	Displays only Write packets in Activity area.
	Polling
	Displays only Polling packets in Activity area.
Device	All
	Displays the information of Local HMI and PLC.
	Local HMI
	Displays the information of Local HMI.
	PLC
	Displays the information of PLC.
Station	Selects the PLC station number to be displayed.
	(This function is disabled when selecting [All] in [Device].)
Address	Selects all or a preferred address type to be displayed.
Туре	(This function is disabled when selecting [All] in [Device].)
Keyword	Displays information that contains the keyword entered.

When communication error occurs, information relating to the error will be highlighted in red in the Activity page, and the cause of error is shown in the Result column.

Object	De	vice P	acket Macr	o MQT	Ī				
Activity	Pollin	g							
Packet Ty	oe: All		• Device:	All	*				
ddress Ty	oe: All		 Keyword: 						<u> </u>
No.	Туре	PID	Device	Station	Address	Length	Index	Time	Result
231	Р	200063	Local HMI		PLW-8950	1		0	Success
230	Р	200062	Local HMI	6	LB-8999	1	e e	17	Success
229	Р	200067	Local HMI	2	LB-12360	1	-	17	Success
228	Р	200065	Local HMI	-	LW-0	150	-	34	Success
227	Р	200063	Local HMI	5	PLW-8950	1	5	0	Success
226	Р	200064	Local HMI	2	LB-0	1	6	16	Success
225	Р	200062	Local HMI	2	LB-8999	1	-	16	Success
224	Р	200066	DELTA DVP	1	D-0	1	-	1087	Ignore
223	Р	200067	Local HMI	18	LB-12360	1		33	Success
222	Р	200063	Local HMI	2	PLW-8950	1	6	0	Success
221	D	200065	Local HMI	12	110/-0	150	1	16	Success

 The Polling page shows the information of the polling packets periodically sent from HMI, including the start address, length...etc.



cMT Diagnoser

Object	Device	Packet	Macro	MQTT				
Activity	Polling							
Packet ID	Device				Station	Address	Length	Index
100338	Local HMI					LB-0	1	
100339	Local HMI				15	LW-0	51	
200062	Local HMI				-	LB-8999	1	
200063	Local HMI				15	PLW-8950	1	
200064	Local HMI				-	LB-0	1	
200065	Local HMI				5	LW-0	150	
200066	DELTA DVP				1	D-0	1	
20006/	Local HMI				5	LB-12360	1	

Item	Description
Packet ID	Use the Packet ID to find the invalid object.
Device	Displays HMI and PLC type.
Station	Displays PLC station number.
Address/Length	Displays the device type and the size of the packet.
Index	Displays the index register number of the object.

31.3.4. Macro Settings

1. Execute and test macros.

D	Name	Execute
000	macro_plus1	0
001	macro_plus10	0
002	macro_plus100	0
003	macro_plus_x	0
004	macro_data_type	0
005	Debug Demo	0
4:36:49] S	rver connected.	

2. Macro Debugger Tool.



cMT Diagnoser

Diagnoser							
Object	Device	Packet	Macro	JS	MQTT	OPCUA	
Execute	Debug						
➡ macro_0		• • •	II C	+ 1	0	Q Search	< >
4 GetDat 5 value= 6 SetDat 7 TRACE(8 end ma	a(value, "1 value+100 a(value, "1 "end macro_ cro_command	Local HMI" Local HMI" _plus100") d	, LW, O,	1)			
Watch				(Dutput		
Name	١	/alue		Туре	[11:30:46] [11:30:46]	Connecting to server Server connected.	

ltem	Description						
Load	Macro Debug File can be generated when compiling a						
macro	project file in EasyBuilder Pro, and it is placed in the same						
debug file	directory as the compilation file. Macro Debug File has a						
	file name extension of ".debug". When an unmatched						
	debug file is loaded, the following message window						
	shows.						
	🖸 cMT Diagnoser						
	This debug file cannot be used with current project.						
	UK						
Start	Start debugging or execute macro from a breakpoint						
debugging	Start debugging of execute macro nom a breakpoint.						
Stop	Stop dobugging						
debugging							
Break	Pause macro debugging manually. A Breakpoint can be set						
	at a statement by clicking the grey margin on the left-hand						
	side of the statement. When a macro is launched, it will						
	run until the Breakpoint, and wait for the next command.						
Restart	After running on-line / off-line simulation in macro work						
simulation	space, the macro can be revised directly in the Diagnoser						
	window. After revising the macro, the simulation can be						
	restarted by clicking Restart Simulation, and then the						
	following confirmation window pops up.						
Restart simulation	side of the statement. When a macro is launched, it will run until the Breakpoint, and wait for the next command. After running on-line / off-line simulation in macro work space, the macro can be revised directly in the Diagnoser window. After revising the macro, the simulation can be restarted by clicking Restart Simulation, and then the following confirmation window pops up.						



Step IntoRun the macro line by line. If the line contains a function, the diagnoser will enter the called function and continue line-by-line debugging from there.Stop OutReturn to the line where the current function was called.Step OverRun the macro line by line. If the line contains a function, the function will be executed and the result is returned without debugging each line.SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.		CMTViewer
Step IntoRun the macro line by line. If the line contains a function, the diagnoser will enter the called function and continue line-by-line debugging from there.Stop OutReturn to the line where the current function was called.Step OverRun the macro line by line. If the line contains a function, the function will be executed and the result is returned without debugging each line.SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.		Macro will be saved and compiled before restarting simulation. Continue?
Step IntoRun the macro line by line. If the line contains a function, the diagnoser will enter the called function and continue line-by-line debugging from there.Stop OutReturn to the line where the current function was called.Step OverRun the macro line by line. If the line contains a function, the function will be executed and the result is returned 		Yes No
the diagnoser will enter the called function and continue line-by-line debugging from there.Stop OutReturn to the line where the current function was called.Step OverRun the macro line by line. If the line contains a function, the function will be executed and the result is returned without debugging each line.SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.	Step Into	Run the macro line by line. If the line contains a function,
line-by-line debugging from there.Stop OutReturn to the line where the current function was called.Step OverRun the macro line by line. If the line contains a function, the function will be executed and the result is returned without debugging each line.SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.		the diagnoser will enter the called function and continue
Stop OutReturn to the line where the current function was called.Step OverRun the macro line by line. If the line contains a function, the function will be executed and the result is returned without debugging each line.SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.		line-by-line debugging from there.
Step OverRun the macro line by line. If the line contains a function, the function will be executed and the result is returned without debugging each line.SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.	Stop Out	Return to the line where the current function was called.
the function will be executed and the result is returned without debugging each line. Search Search for a macro by entering keywords. Watch When debugging, variables can be added here for watching the change of data. Output Shows system information or TRACE output.	Step Over	Run the macro line by line. If the line contains a function,
without debugging each line.SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.		the function will be executed and the result is returned
SearchSearch for a macro by entering keywords.WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.		without debugging each line.
WatchWhen debugging, variables can be added here for watching the change of data.OutputShows system information or TRACE output.	Search	Search for a macro by entering keywords.
watching the change of data.OutputShows system information or TRACE output.	Watch	When debugging, variables can be added here for
Output Shows system information or TRACE output.		watching the change of data.
	Output	Shows system information or TRACE output.

Macro Debug in cMT Diagnoser can be opened in the following ways, each with different privilege requirements and operation steps.

Way1. [On-line Debug] or [Off-line Debug] within macro editing work space.

e execution condition ecute one time when HMI starts Password Protect [Disable]
ecute one time when HMI starts Password Protect [Disable]
Password Protect [Disable]

Click [On-line Debug] or [Off-line Debug]. Software will generate a Macro Debug File. Also,





macro editing work space window will close and be replaced by cMT Viewer simulation and cMT Diagnoser, the latter of which loads the Macro Debug File automatically. When the macro is being debugged in cMT Diagnoser's, the macro codes can actually be modified.

Upon completing macro debugging, close the cMT Viewer simulation. The macro editing work space should appear again, and it will ask whether you want to update the macro with the changes made in cMT Diagnoser.

Way2. On-line simulation / off-line simulation in EasyBuilder Pro.

The Macro Debug File is automatically generated after compiling the project file. When cMT Viewer simulation window is shown by running simulation in EasyBuilder Pro, open cMT Diagnoser from the right-click menu. The system imports the debug file automatically when it detects one. In simulation mode, macro can only be debugged but not changed.

Note: There must be at least one macro in the project for the Macro Debug File to be generated during compilation.

Way3. Downloading the project file to HMI and then running cMT Diagnoser. The Macro Debug File should be manually imported in the Macro Debugger, and under this mode the macro can only be debugged but not changed.

An Example of Macro Debugging

The following steps show an example about how to debug macro.

1. Click [Start Debugging].



🖬 cMT Diagnoser					
Cobject Device Packet Macro	MQTT				ŝ
Execute Debug					
🛃 Debug Demo 🔹 🕨 🔳 🔢 🚦 🕄	•		Q Search	<	>
<pre>1 2 sub int Func(int i) 3 3 4 int h=9 i=9*h 5 6 end sub 7 8 macro_command main() 9 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1</pre>	6,7,8,9	,10}			
Watch		Output			
Name Value	Туре	[14:57:23] Connect [14:57:23] Server o	ting to server connected.		

2. Set Breakpoints in the grey margin on the left-hand side of the lines. The Breakpoints are represented by red dots.

🖸 cMT Diagnoser		_ • ×
< Object Device Packet Macro	ΜQTT	Ś
Execute Debug		
🛃 Debug Demo 🔹 🕨 🔲 🕴 😘	Search	< >
<pre>1 2 sub int Func(int i) 3 int h=9 4 i=9*h 5 end sub 7 8 macro_command main() 9 int a=0, b=1 10 int array[10]={1,2,3,4,5, 11 12 for a= 0 to 5 step 1 13 b = b+a 14 next a 15 16 int c=0 17 c = Func(b)</pre>	6,7,8,9,10}	
Watch	Output	
Name Value	Type [14:57:23] Connecting to server [14:57:23] Server connected.	

3. After debugging is started, it will stop at the first breakpoint, now marked by a yellow arrow.



🗈 cMT Diagnoser					
≺ Object De	vice Packet Mac	ro MQTT			
Execute Debug	g				
L Debug Demo	• • • • •	÷ 🕈	Q Sea	arch	< >
1 2 3 4 5 6 1 7 8 8 8 10 12 12 13 14 15 16 17 17 16 17 17 16 17 16 17 16 16 17 17 17 16 16 16 17 16 16 16 16 17 16 16 16 17 16 16 16 16 16 16 16 16 16 16	<pre>Func (int i) int h=9 i=9*h return i mmand main() int a=0, b=1 int array[10]={1,2,3 for a= 0 to 5 step 3 b = b+a next a int c=0 c = Func(b)</pre>	3,4,5,6,7,8,9 L	,10}		
Watch			Output		
Name	Value	Туре	[17:03:12] Connecting to s [17:03:13] Server connect [17:03:19] Execute macro [17:03:19] [ID 002, Line 2] [17:03:19] [ID 002, Line 2] [17:03:53] [ID 002, Line 2] [17:03:53] [ID 002, Line 2] [17:03:54] [ID 002, Line 7] [17:04:20] Execute macro	erver td. (ID 002) begin macro_plus100 end macro_plus100 (ID 002) begin macro_plus100 end macro_plus100 (ID 005)	

At this moment, variables can be added monitored in the Watch table as shown below.
 Contents of arrays can be monitored as well.

🖸 cMT Diagnoser				- (
< Object Devic	e Packet Macro	MQTT			ŝ
Execute Debug					
Debug Demo	• • • • • • •	÷	◯ Search	<	>
1 2 sub int Fun 4 i= 5 re 6 end sub 7 8 macro_comma 9 9 9 in 10 in 11 12 fo 13 14 ne 15 in 17 c	<pre>ac(int i) it h=9 i9*h iturn i and main() it a=0, b=1 it array[10]={1,2,3,4,5, or a= 0 to 5 step 1</pre>	6,7,8,9	,10}		
Watch			Output		
Name	Value 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	Type int int int int int int int int	[17:03:19] Execute macro (ID 002) [17:03:19] [ID 002, Line 2] begin macro_plus100 [17:03:19] [ID 002, Line 7] end macro_plus100 [17:03:53] [ID 002, Line 7] begin macro_plus100 [17:03:54] [ID 002, Line 7] begin macro_plus100 [17:03:54] [ID 002, Line 7] end macro_plus100 [17:04:20] Execute macro (ID 005) [17:10:32] Execute macro (ID 005) [17:11:40] Execute macro (ID 005) [17:11:40] Execute macro (ID 002)		

5. After the Start Debugging button is clicked once again, macro will run and stop at the next breakpoint. The changes of values are shown in the Watch list.



🖸 cMT Diagnoser					• X
< Object Devic	e Packet Macro	MQTT			
Execute Debug					
Debug Demo	• • • • • • •	•	Q Search	<	>
1 2 sub int Fur 4 5 re 6 end sub 7 8 macro_comma 9 jir 10 ir 11 12 fc 13 14 ne 15 re 14 ne 17 c	<pre>http://docs.org/line/action/acti</pre>	6,7,8,5	€,10}		
Watch			Output		
Name	Value 0	Type int	[17:03:12] Connecting to server [17:03:13] Server connected.		
 ▶ array ♦ b 	1, 2, 3, 4, 5, 6, 7, 8, 9, 10 1	int int	[17:03:19] Execute macro [ID 002] [17:03:19] [ID 002, Line 2] begin macro_plus100 [17:03:19] [ID 002, Line 7] end macro_plus100 [17:03:53] Execute macro [ID 002] [17:03:53] [ID 002, Line 2] begin macro_plus100 [17:03:54] [ID 002, Line 7] end macro_plus100 [17:04:20] Execute macro (ID 005) [17:10:32] Execute macro (ID 002)		

6. Click the Step Into button to run macro debugging line by line. If the line contains a function, cMT Diagnoser will enter the called function and continue line-by-line debugging from there. Changes of values can be observed in the Watch table.

cMT Diagnoser						
Object De	vice Packet	Macro	MQTT			ź
Execute Debu	g					
🛃 Debug Demo	• • •	III † 4	2	◯ Search	<	>
• 2 sub int 1 4 5 6 end sub 7 8 macro_con 9 10 11 12 13 14 15 6 0 17 18 end macro	<pre>Func(int i) int h=9 i=9*h return i mmand main() int a=0, b=1 int array[10]== for a= 0 to 5 :</pre>	1,2,3,4,5, step 1 a	,6,7,8, <u>9</u>	9,10}		
Watch				Output		
Name a b c	Value 6 16 0		Type int int int	[17:03:12] Connecting to server [17:03:13] Server connected. [17:03:19] Execute macro (ID 002) [17:03:19] [ID 002, Line 2] begin macro_plus100 [17:03:19] [ID 002, Line 7] end macro_plus100 [17:03:53] [ID 002, Line 7] end macro_plus100 [17:03:53] [ID 002, Line 2] begin macro_plus100 [17:03:54] [ID 002, Line 7] end macro_plus100 [17:04:20] Execute macro (ID 005)		

7. Click the Step Over button to run the macro line by line. Now, if the line contains a function, the result will be returned without debugging each line within the function call. When debugging meets a Breakpoint within the function call though, it stops there.



💶 cMT Diagnoser				
< Object De	evice Packet Mac	ro MQTT		ŝ
Execute Debu	g			
🛃 Debug Demo	• • • • •	÷ 📀	◯ Search	< >
2 sub int 3 4 5 6 end sub 7 8 macro_cc 9 10 11 12 13 14 15 16 0 17 ◆ 18 end macr	<pre>Func (int i) int h=9 i=9*h return i mmand main() int a=0, b=1 int array[10]={1,2,3 for a= 0 to 5 step 3 b = b+a next a int c=0 c = Func(b) o_command</pre>	3,4,5,6,7,8,9 1	,10}	
Watch			Output	
Name a b c	Value 6 16 81	Type int int int	[17:41:17] Connecting to server [17:41:17] Server connected. [17:41:43] Execute macro (ID 005) [17:41:59] Execute macro (ID 005) [17:41:59] Execute macro (ID 005)	

8. Click Stop Debugging button to stop.

🖸 cMT Diagnoser			
Cobject Device Pace	ket Macro MQT		çç,
Execute Debug			
▶ Debug Demo ▼	▶ ■ = + ◇	◯ Şearch	< >
<pre>2 sub int Func(int i) 3 int h=9 4 i=9*h 5 return i 6 end sub 7 8 macro_command main 9 int a=0, k 10 int array 11 12 for a= 0 t 13 k 14 next a 15 16 int c=0 17 c = Func(k end macro_command</pre>) () b=1 [10]={1,2,3,4,5,6,7,1 to 5 step 1 b = b+a b)	3,9,10}	
Watch		Output	
Name Value ◆ a 6 ◆ b 16 ◆ c 81	e Type int int int	[17:41:17] Connecting to server [17:41:17] Server connected. [17:41:43] Execute macro (ID 005) [17:41:59] Execute macro (ID 005) [17:41:59] Execute macro (ID 005)	

31.3.5. MQTT Settings

 In Server tab the type and state of MQTT server is shown. Supported cloud services include: Normal / Azure IoT Hub / Sparkplug / Google Cloud IoT Core. Please note that AWS IoT is not supported.



	:MT Diagnoser						
<	Obiect	Device	Packet	Macro	мотт		
Ľ	0.01000	000000	T d dittot	maaro			-0-
	Server	Address	Published	Subscribed			*
	Type: Norm	al					
	State: Conne	cted					
	[10:29:01] CI [10:29:01] CI [10:29:12] CI [10:29:12] CI [10:29:21] CI [10:29:21] CI [10:29:21] CI [10:29:31] CI [10:29:31] CI [10:29:42] CI [10:29:51] CI [10:29:51] CI [10:30:01] CI [10:30:01] CI [10:30:01] CI [10:30:21] CI [10:30:21] CI [10:30:21] CI [10:30:21] CI [10:30:21] CI [10:30:21] CI [10:30:21] CI [10:30:51] CI [10:31:01] CI [10:31:101] CI [10:31:12] CI	ent 407d7a6 ent 407d7a6	c-8ff8-4e59-b c-8ff8-4e59-b	36f-2b3d661 36f-2b3d661	ed918 sending PINGREQ ed918 received PINGRESP ed918 sending PINGREQ ed918 sending PINGREQ ed918 sending PINGREQ ed918 received PINGRESP ed918 sending PINGREQ ed918 received PINGRESP ed918 sending PINGREQ ed918 sending PINGREQ ed918 sending PINGREQ ed918 received PINGRESP ed918 sending PINGRESP ed918 sending PINGREQ ed918 received PINGRESP ed918 sending PINGREQ ed918 received PINGRESP ed918 sending PINGRESP ed918 sending PINGRESP ed918 sending PINGRESP ed918 sending PINGRESP		8

2. In Address tab the Topics published and subscribed by the HMI are shown. Selecting the checkbox of a topic in this tab can make cMT Diagnoser receive all MQTT messages published and subscribed, allowing cMT Diagnoser to change address data for easier MQTT debugging. Only normal servers are supported.

MT Diagnoser						
Object Device	Packet	Macro	MQTT			
Server Address	Published	Subscribed				
Торіс	Device	Address	Туре	Length	Value	
 Publisher 						
I topic 1					Test/MQTT	
Bit value	Local HMI	LB-0	Bit	1	false	
Word value	Local HMI	LW-0	16-bit Uns	1	300	
topic 2					Sub/MQTT	
Data1	Local HMI	LW-30	16-bit Uns	1	0	
Data2	Local HMI	LW-40	16-bit Uns	1	0	
Data3	Local HMI	LW-50	16-bit Uns	1	0	
 Subscriber 						
topic 1					Test/MQTT	
Bit value	Local HMI	LB-2000	Bit	1	false	
Word value	Local HMI	LW-2000	16-bit Uns	1	300	
topic 2					Sub/MQTT	
Data1	Local HMI	LW-1000	16-bit Uns	1	0	
Data2	Local HMI	LW-1001	16-bit Uns	1	0	
Data3	Local HMI	LW-1002	16-bit Uns	1	0	

3. In Published tab the message published from HMI is shown. When Raw Data is used as content format, data will be shown in hexadecimal. Only normal servers are supported.





4. In Subscribe tab the message subscribed by HMI is shown. When Raw Data is used as content format, data will be shown in hexadecimal. Only normal servers are supported.

	:MT Diagnoser						_ • •
<	Object	Device	Packet	Macro	ΜQTT		રંદુર
	Server Add	lress P	ublished	Subscribed		Q Nickname / 1	Fopic 🤾
	Nickname	Topic				Data length	Timestamp
	topic 2	Sub/MQ	ττ ₂			137 bytes	2020-02-15 11:06:17
	{ "d" : { "Dat "Dat }, "ts" : " }	a1" : [a2" : [a3" : [2020-02-	1], 1], 0] 15T20:06	: 33 . 741354'			

31.3.6. OPC UA Settings

1. Users can log in by entering the credentials to monitor values of OPC UA nodes.



Diagnoser											•
Object	Device	Packet	Macro	M	QTT	OPCUA	_				
State: Connecte	d					User: a	dmin	Password	d: ••••	••	€∋
Browser				С		Monitor				=	Ē
Name			Status			Name	Id	DataType		Value	
 Local HMI Weintek Bu Tags 	ilt-in CODESYS	5				CODES	NS2 Strin	Boolean	false		
CODE ▷ Statistics	SYStag 5		Readable		Þ						

31.3.7. JS Settings

1. In this tab users can view console messages of JS objects.

🖸 Diagnoser								
Object	Device	Packet		JS	MQTT	OPCUA		
								*
[17:11:42] [mu [17:11:42] ma [17:11:43] [mu [17:11:43] mu [17:11:43] mu [17:11:43] [mu [17:11:43] [mu [17:11:44] [mu	vusemove] eve rkers [{ x: 216. pusemove] eve rkers [{ x: 216. 0.0714285:	$\begin{array}{l} \text{ent} = \{x: 258, y\\ 0.071428571428\\ \text{ent} = \{x: 246, y\\ 0.071428571428\\ 71428571428\\ 71428571428\\ 7142858, y: 24\\ \text{ent} = \{x: 282, y\\ \text{ent} = \{x: 314, y\\ \text{ent} = \{x: 386, y\\ \text{ent} = \{x: 436, y\\ \text{ent} = \{x: 436, y\\ \text{ent} = \{x: 437, y\\ \text{ent} = \{x: 433, y\\ 142857142857\\ 142857142857\\ 1570876, y: 25\\ \text{ent} = \{x: 217, y\\ \text{ent} = \{x: 217, y\\ \text{ent} = \{x: 215, y\\ 0.07142857142857\\ 142857857\\ 142857857\\ 142857857\\ 1428578578\\ 142857857857\\ 142857857857\\ 142857857857857857\\ 142857857857857857857857\\ 1428578578578578578578578578578578578578578$: 223 } 558, y: 220.1(: 228 } : 248 } : 248 } : 248 ; : 221 1. : 214 } : 214 } : 214 } : 222 ; : 225 ; : 244 ; : 222 ; : 222 ; : 222 ; : 225 ; : 224 ; : 225 ; : 225 ; : 243 ; : 17, y: 243.2; : 17, y: 243.2; : 17, y: 243.2; : 5:55555555; : 264 ; : 266 ; : 267 ; :	0837438 0837438 124845 } 9285714 8991596 8163265 8163265 55557 } 28557142	423644 }] 423644 },] 285714 }] 638656 }] 30612 },] 857144 }]			

31.4. Additional Notes

- 1. cMT Diagnoser supports the monitoring all PLC addresses that HMI can access, including Tag PLC addresses.
- 2. Address values can be displayed in Decimal/HEX/Binary formats. Right-click anywhere on a monitor page to bring the menu up for the change.
- 3. When closing cMT Viewer with cMT Diagnoser opened, cMT Diagnoser will resume last state and pops up when cMT Viewer simulation window opens again. If Diagnoser is



closed before closing cMT Viewer, users should manually open Diagnoser in next simulation.

- 4. When monitoring cMT-SVR/cMT-SVRX models, connecting directly with cMT Diagnoser displays only global object information. To view information for objects within the current window, connect through cMT Viewer and access Diagnoser with a right-click.
- 5. When [Password Protect] is enabled in Macro Manager, the Macro Debug File cannot be generated after compilation.
- 6. Password Protect can be enabled for an individual macro, and two modes are available: Encrypted and Read Only. To debug a password protected macro, a password is required to unlock the macro in cMT Diagnoser. If the macro is protected in Read Only mode, it can only be viewed but not changed when debugging. If the macro is protected in Encrypted mode, it cannot be viewed at all.
- 7. When a macro is not successfully compiled, it will be put into the Macro Under Development list and will not be contained in the Macro Debug File.
- 8. cMT Diagnoser can only debug one macro at a time. When debugging a macro, executing another macro will not be effective.
- 9. When the breakpoint is placed on irrelevant lines, such as beside an empty line, variable declaration without initial value assignment, the breakpoint will be represented by a hollow red dot as line 7 below shows.

Diagnoser				_ • • ×
Object	Device Packet	Macro MC	זק	
Execute	Debug			
上 Debug Dem		II + 🔿 👘	◯ Search	< >
2 sub 3 4 5 6 end 3 7 8 macro 9 10 11 12 13 14 15 16 17 18 end 1	<pre>int Func(int i) int h=9 i=9*h return i sub p_command main() int a=0, b=1 int array[10]={1, for a= 0 to 5 step b = b+a next a int c=0 c = Func(b) macro_command</pre>	2,3,4,5,6,7,8, p 1	9,10}	
Watch			Output	
Name	Value	Туре	[11:20:18] Connecting to server [11:20:18] Server connected.	



31-20

