SRS

XMPS-1000

(GUI for programming software of PLC)

Version<3.0>

1. **Graphical UI of programming software**
   Ladder, Predefined FB, user defined FB etc.
2. **Interpreter (Application program)**
3. **Interpreter (PLC configuration settings)**

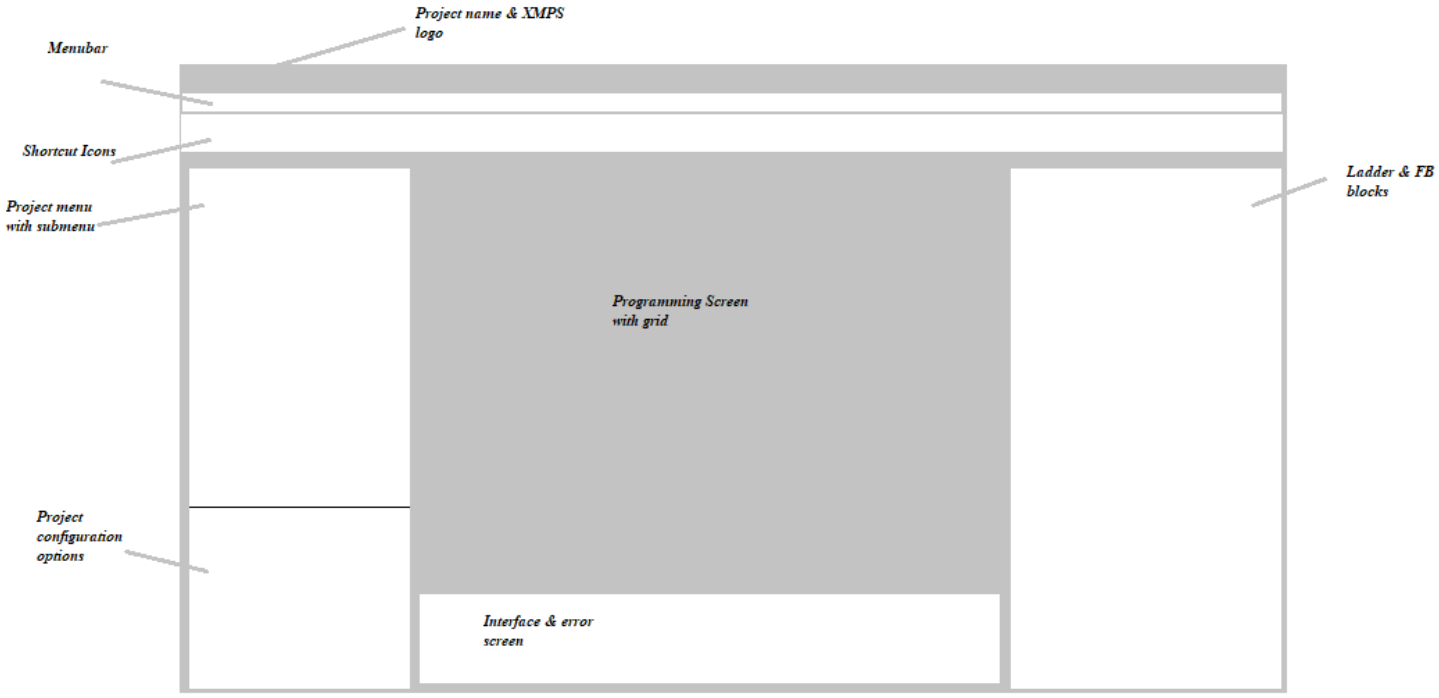
1. Introduction :

This is the Software Requirements Specification (SRS) document which provides an overview of the entire XMPS-1000 requirements.

2. Purpose:

In short, the purpose of this SRS document is to provide a detailed overview of our XMPS- 1000 software product, its parameters and goals. This document describes the project's user interface requirements. It will explain the purpose and features of the system, the interfaces of the system, Interpreter (code generation).

Provisional Proposed UI :



4. Menu Bar :

4.1. Project : In “Project” menu following options should be added.

4.1.1 New Project

Frontend-New Project

Backend – a. New path for project

b. Select CPU option

c. If existing path choosed popup of warning.

4.1.2 Open Project

Frontend- Open Project

Backend- a. Open project popup

b. If exixsting project not saved then popup for saving the project

4.1.3 - Save project

Frontend- Save project

Backend- a.Save project latest changes

4.1.4 Save As..

Frontend-Save As..

Backend- a.Ask for saving project path

4.1.5 Close Project

Frontend- Close Project

Backend- a. Save the latest changes and close the project

b. only project should close not programming software

4.1.6 Print

Frontend- Print

Backend- a. Same as what we have done in XMPS-100

Print the ladder main program & configuration settings.

4.1.7 Exit

Frontend-Exit

Backend- a.Save and close the software.

4.2 Edit : In “Edit” menu following options should be added.

4.2.1 Undo

Frontend-Undo

Backend- a. Undo the latest change (atleast last 15 changes)

4.2.2 Redo

Frontend- Redo

Bacckend- a. Redo the changes (atleast 15 changes)

4.2.3 Copy

Frontend-Copy

Backend- a. Copy the selescted rung/instruction

b. Copy the selected project folder

c. Copy the selected program logic file

4. 2.4 Paste

Frontend-Paste

Backend- a.Paste the copied /cutted rung, instruction

b. Paste the copied/cutted folder

c. Paste the copied program logic file

4. 2.5 Cut

Frontend-Cut

Backend- a.Cut the selected rung/ instruction

b. Cut the folder

c. Cut the Program logic file

4. 2.6 Delete

Frontend-Delete

Backend- a.Delete the selected rung/ instruction

b. Delete the slecetd folder

c. Delet the selected program logic file

4.2.7 Find & Replace

Frontend-Find & Replace

Backend- a. Find the text

b. Fing & replace the text

4.3 View : In “View” menu following options should be added.

4.3.1 Device info

Frontend-Device info

After Click- It will show the XMPRo CPU parameters

4.3.2 Zoom

Frontend-Zoom

Backend-Should able to zoom the programming grid window

4.3.3 Project Window

Frontend-Project Window

Backend-After click project window should Enable or Disable

4.3.4 Compiler error screen

Frontend-Compiler error screen

Backend-After click compiler screen should Enable or Disable

4.4 Mode : In “Mode” menu following options should be added.

4.4.1 Login

Frontend- Login

Backend-Sotware should connect with PLC via ethernet port & all editable functions should be disable, no program will edit in this mode only start & stop plc command should accept.

Online Monitering option should enable.

4.4.2 Logout

Frontend-Logout

Backend- Logout the Login mode and Normal mode activate.

4.4.3 Download Project

Frontend- Download Project

Backend-Save - Compile - Download the compiled file into the PLC via ethernet port

4.4.4 Upload Project

Frontend- Upload project

Backend- Upload the project from PLC & display the project

4.4.5 Offline Simulation

Frontend-Offline simulation

Backend- TBD

4.4.6 PLC Start

Frontend- PLC start

Backend-PLC start command shoud go to PLC & this mode should active only if the user is Login to PLC

4.4.7 PLC Stop

Frontend- PLC stop

Backend-PLC stop command shoud go to PLC & this mode should active only if the user is Login to PLC

4.4.8 Compile

Frontend- Compile

Backend- Generate code as per requirement.

4.5 Help : In “Help” menu following options should be added.

4.5.1 Index

Frontend Index

After click the Index of XMPS-1000 intruction help window should be open. And after clicking each of index point it will show the detail information of perticuler point.

4.5.2 Contents

Frontend-Contents

After click the Contents all help related to XMPS-1000 software should open.

4.5.3 Search

Frontend-Search

Should open the search bar for the user.

4.5.4 User amnual

User manual PDf open.

5. Shortcut Bar :

In shortcut bar following symbols should be present:

5.1 New
5.2 Open
5.3 Save
5.4 Project close
5.5 Upload
5.6 Download
5.7 Zoom IN
5.8 Zoom out
5.9 Zoom % selection
5.10 Compile
5.11 Login
5.12 Logout
5.13 Run Online Monitering
5.14 Help
5.15 Cut
5.16 Copy
5.17 Paste
5.18 Select
5.19 Undo
5.20 Redo
5.21 Delete
5.22 Previous screen
5.23 Next screen
5.24 Find
5.25 Ladder components (Contact,Coil,Parallel contact,Parallel coil, Variable,New rung,New comment)

After click on perticuler shortcut related action should be taken.

6. Project Menu

In Left side of screen there should be the Project Menu with the Submenus as follows:

-All files

6.1 Project List

6.1.1 Project name XXXXXX

6.1.1.1 Powerup routine (Init)

6.1.1.1.1 Logic 1

6.1.1.1.2 Logic 2

6.1.1.1.3 xxxxxx

6.1.1.2 Main

6.1.1.2.1 Main program (should call the Logic blocks as per sequence)

6.1.1.3 Library

6.1.1.3.1 Logic blocks

6.1.1.3.1.1 Logic 1

6.1.1.3.1.2 Logic 2

6.1.1.3.1.3 Logic 3

6.1.1.3.1.4 Logic 4

6.1.1.3.1.5 xxxxxx

6.1.1.3.2 Hardware Interrupt

6.1.1.3.2.1 Logic 1

6.1.1.3.2.2 Logic 2

6.1.1.3.3 UDFB

6.1.1.3.3.1 FB 1 xxxxx

6.1.1.3.3.2 FB 2 xxxxx

6.1.1.3.3.3 FB 3 xxxxx

6.1.1.4 IO configuration

6.1.1.4.1 Base (XMPRO-10)

6.1.1.4.2 Local IO 1 (xxxx)

6.1.1.4.3 Local IO 2 (xxxx)

6.1.1.4.4 Local IO 3 (xxxx)

6.1.1.4.5 Local IO 4 (xxxx)

6.1.1.4.6 Local IO 5 (xxxx)

6.1.1.5 Tags

6.1.1.6 Error diagnostic tags

6.1.1.7 System Configuration

6.1.1.7.1 Ethernet

6.1.1.7.1.1 Modbus TCP Server

6.1.1.7.1.2 Modbus TCP Client

6.1.1.7.1.2.1 Slave 1
6.1.1.7.1.2.2 Slave 2
6.1.1.7.1.2.3 Slave 3
6.1.1.7.1.2.4 xxxxxx

6.1.1.7.2 RS485

6.1.1.7.2.1 Modbus RTU Master

6.1.1.7.2.1.1 Slave 1 (MOD-DI-8)
6.1.1.7.2.1.2 Slave 2 (xxxxx)
6.1.1.7.2.1.3 Slave 3 (xxxxx)
6.1.1.7.2.1.4 Xxxxx

6.1.1.7.3 CAN

6.1.1.7.3.1 CANOpen Master

6.1.1.7.3.1.1 CANOpen Slave 1 (CAN -DI16)
6.1.1.7.3.1.2 CANOpen Slave 2
6.1.1.7.3.1.3 CANOpen Slave 3
6.1.1.7.3.1.4 xxxxxxx

6.1.2 Project name XXXXXX

Same as project 1 …….

All tabs should include the dedicated screen as defined below.

Project name Screen- After click it should display the overall program information

(No.of Power up blocks, No of Logic blocks,Total IO used,Total tags defined etc)

Powerup routine (Init) Screen- Under this tab Logic blocks should be added as per requirement of user.

Main- Under this tab one “Main Program” block should add. Here user can call the Logic blocks as per their sequence. (It will be the programming grid screen)

Library- In this tab user should able to select the which following library he wants to use for his application.

Logic blocks-Under this tab user should add the Logic blocks as per their requiement and should rename that block as per their requirement.

Hardware interrupt- In this tab user should configure the Interrupt Input and Under that Input tab user should add the Logic blocks which should run when Interrupt is occure.

UDFB- User defined function block. Under this tab user can create his own programmed function block.

Function block has Inputs & outputs, so user will able to add the Inputs & can create his own logic as per received input and gives the Output.

IO configuration- Under this tab user should able to configure the Onboard and Local IO modules.

Tags- In this tab all pregenerated CPU & Local IO tags should be automatically assigned as per predefined address.

User can able to add the memory tags as per their requirement in Logic block screen only and same tag should add in Tags screen automatically.

User should able to rename the tag name only.

User should able to add the memory address tags as per their requirement.

All memory tags which will generate in all logic blocks during the program should add here automatically.

Error Diagnostic tags- In this tab all predefined error tags will be added as per added configuration.

System Configuration- Under this tab all settings and requests of Ethernet,Modbus TCP Server,Modbus TCP Client,RS485,Modbus RTU masterSlave 1,Slave 2,Slave xxx ,CAN, CANOpen master,CANOpen slave 1, CANOpen slave xx should be added.

7. Ladder & FB Blocks screen

This screen should be activate when user will click on any Logic block screen.

Under this screen all below instructions predefined FB should be present.

All Ladder components should be present here.

All UDFB blocks also should be added here automatically when user creates any UDFB.

A. Ladder components:

7.1 Contact
7.2 Coil
7.3 Insert after contact
7.4 Parallel Contact
7.5 Insert FB
7.6 Variable
7.7 Set Coil
7.8 Reset Coil
7.9 Negate Contact
7.10 Negate Parallel contact
7.11 Comment
7.12 Insert branch

B. Predefined Function blocks:

7.13 Logical
- 7.13.1 AND
- 7.13.2 OR
- 7.13.3 XOR
- 7.13.4 NOT

7.14 Arithmatic
- 7.14.1 ADD
- 7.14.2 SUB
- 7.14.3 MUL
- 7.14.4 DIV
- 7.14.5 MOD
- 7.14.6 MOV

7.15 Bitshift
- 7.15.1 SHL
- 7.15.2 SHR
- 7.15.3 ROR
- 7.15.4 ROL

7.16 Limit

7.17 Compare
- 7.17.1 GT
- 7.17.2 GE
- 7.17.3 LT
- 7.17.4 LE
- 7.17.5 EQ

7.17.6 NE

7.18 Edge detector

7.18.1 Rising edge

7.18.2 Falling edge

7.19 Counter

7.19.1 CTU

7.19.2 CTD

7.20 Timer

7.20.1 0.01S TON

7.20.2 0.1S TON

7.20.3 1S TON

7.20.4 0.01S TOFF

7.20.5 0.1S TOFF

7.20.6 1S TOFF

7.20.7 0.01S TP

7.20.8 0.1S TP

7.20.9 1S TP

7.21 Flipflop

7.21.1 RS

7.21.2 SR

8. Interface & Error screen

8.1 This screen will appear when user will build or compile the project and shows the status of compilation. If any error is present then it should show the error.

(Next version of this document will explain the details about this screen).

9. Programming grid screen

In this screen user can drag and drop the all ladder components & function blocks.

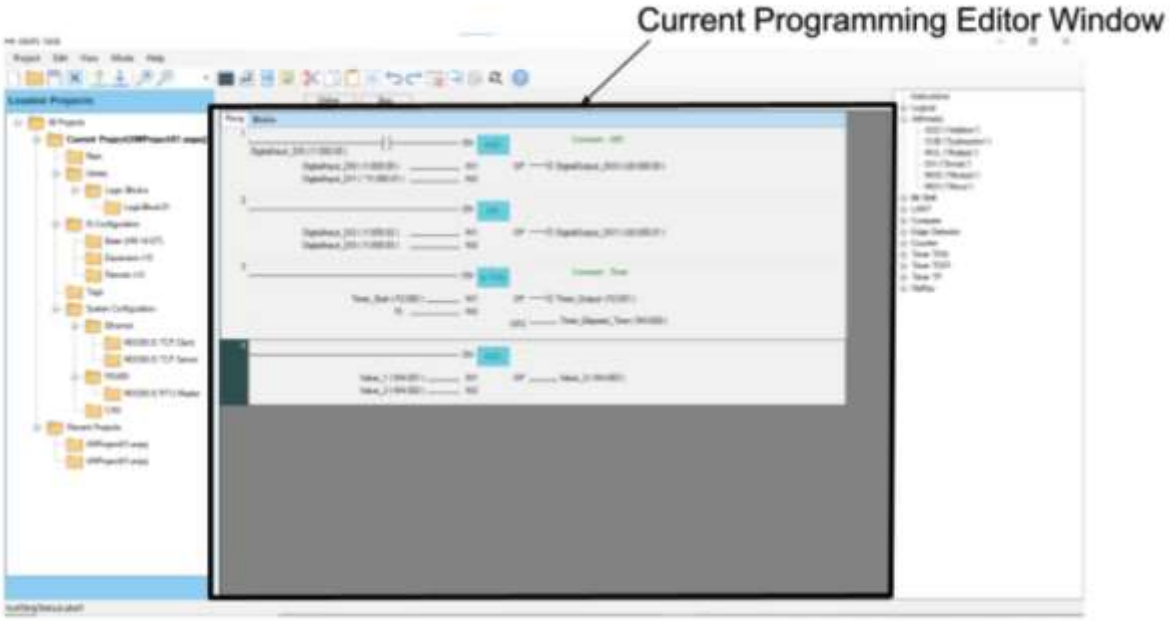
Screen grid should be adjustable.

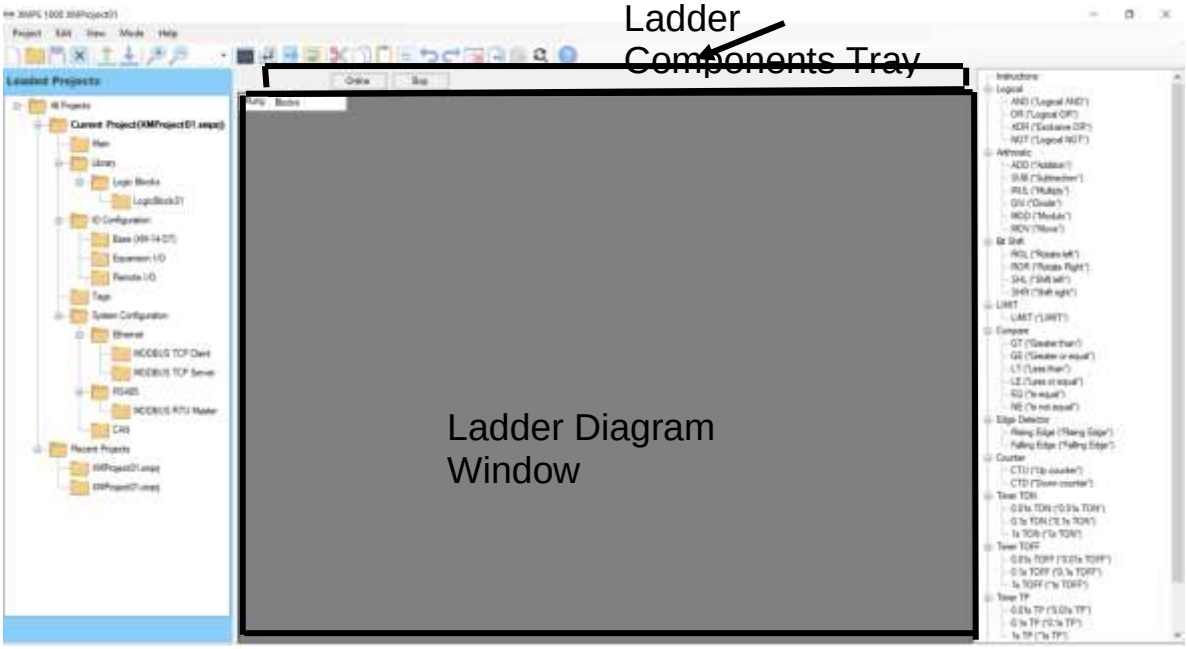
Presently we will limit the 8 ladder components in one rung.

And same for parallel also, we will limit this to 8 rungs.

Use the right click & double click for adding & displaying the ladder components & its information.

XMPS-1000 Software Programming grid screen





As seen in the image above two things are missing -

1. **Ladder & Block Components Tray or Window**
2. **Ladder Diagram Window**

In this screen user can drag and drop or select the rung and click the shortcut of any ladder components & function blocks as desired.

Use the right click & double click for adding & displaying the ladder components & its information.

Screen grid should be adjustable.

Presently we will limit the 12 ladder components in one rung.

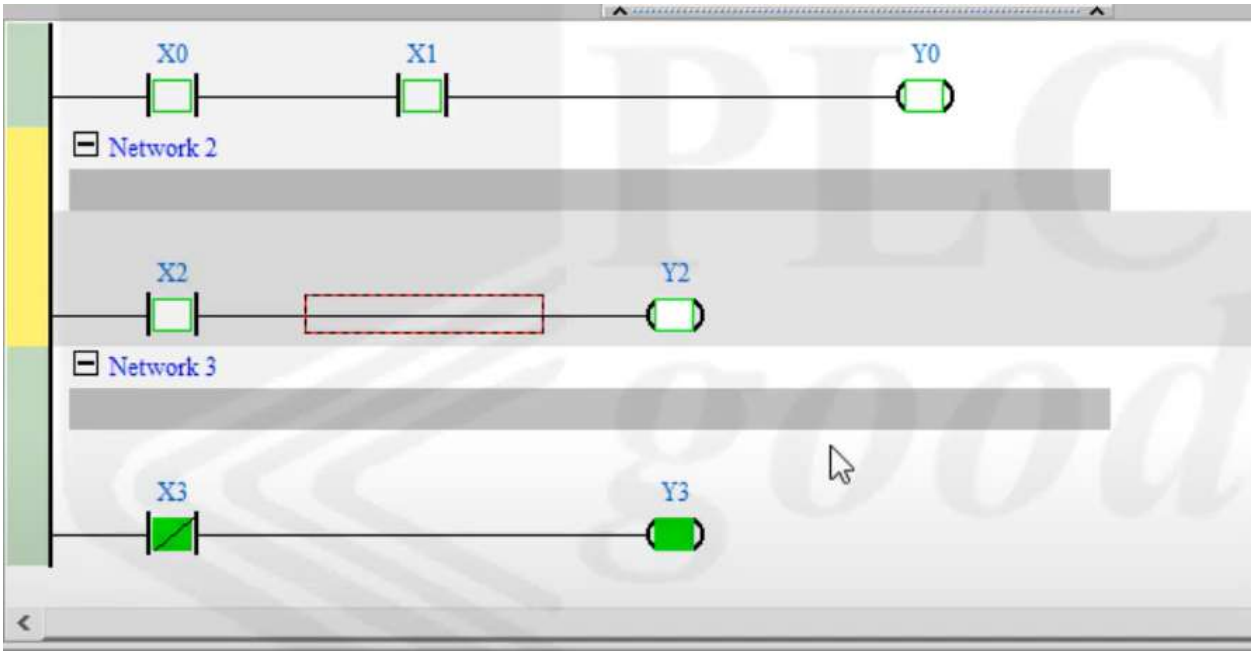
And same for parallel also, we will limit this to 12 rungs.

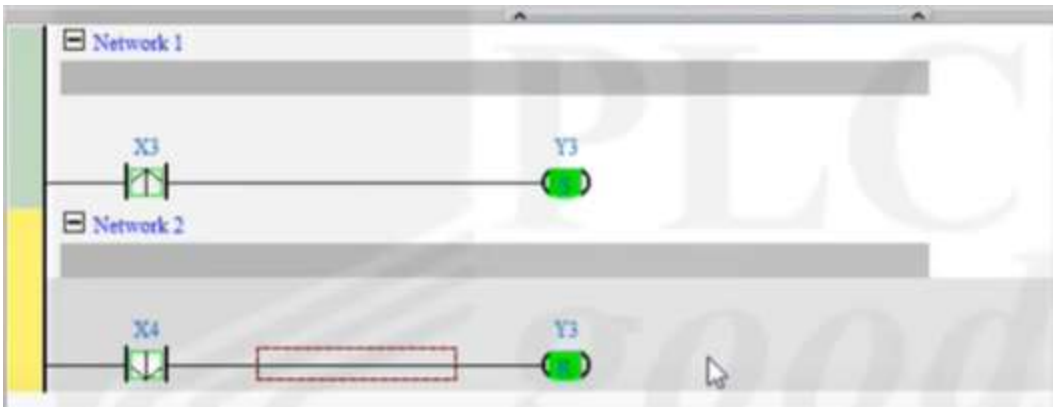
10. Examples of Ladder Diagram screens of other PLC makes softwares

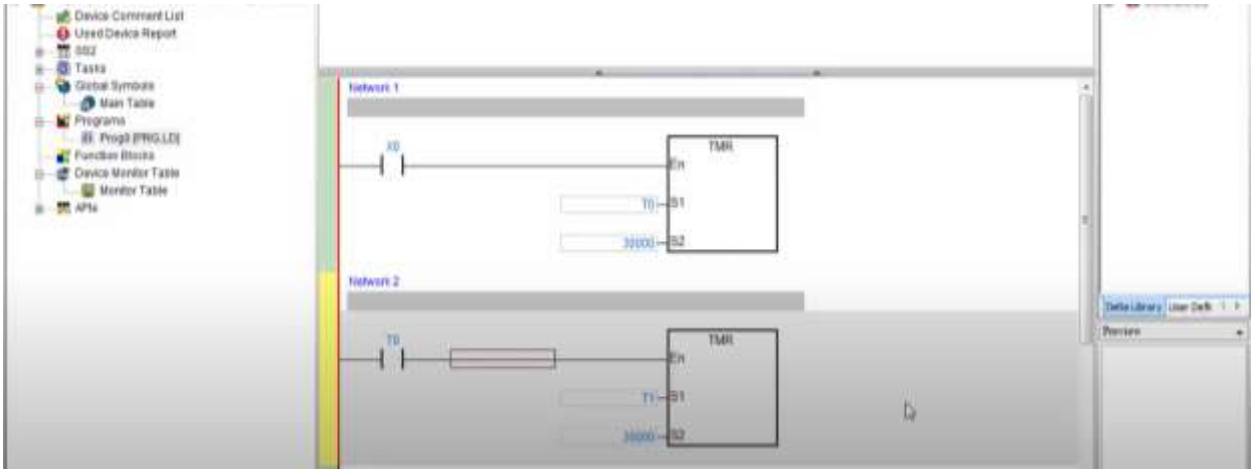
Following are the Examples of Delta & Fatek PLC software with YouTube link.

- https://www.youtube.com/watch?v=TxgxGx8JMeI&list=PLRLTfp5JUl6Q80pK_apPaHXdgetnzXFd6&index=3
- https://www.youtube.com/watch?v=ARhdWtmGhh8&list=PLO_z-Qzju3sdeoAi5XLRxy9b6ZItUx1nz&index=14

9.2 DVP Soft (DELTA)

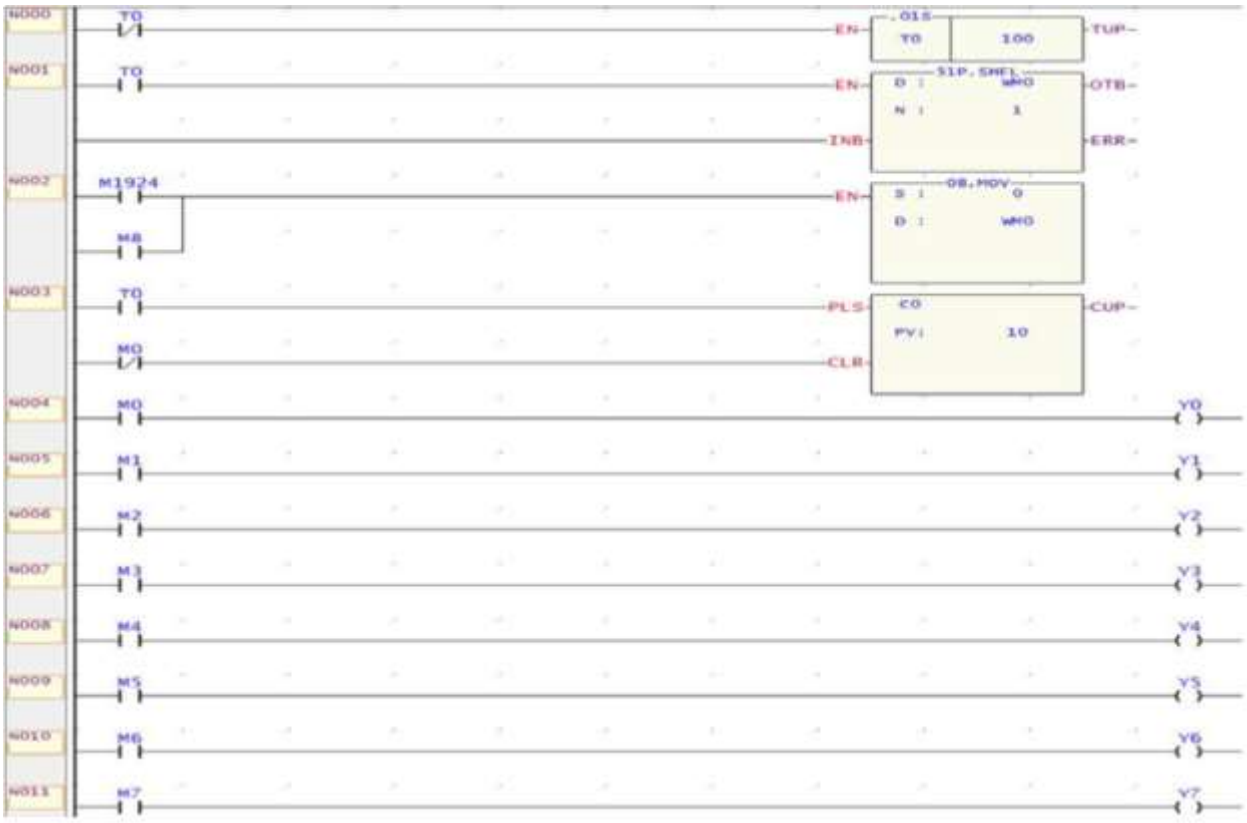



Network 1
X3
Y3
Network 2
X4
Y3


Device Comment List
Used Device Report
SD2
Tasks
Global Symbols
Main Table
Programs
Prog0 [PRG,LD]
Function Blocks
Device Monitor Table
Monitor Table
APIs
Network 1
X0
TMR
En
T0
S1
30000
S2
Network 2
T0
TMR
En
T1
S1
30000
S2

9.1 WinPro Ladder (FATEK)

(Component tray)



10. CPU Addresses:

This is the current CPU addressing structure of XMPS 1000 software but we intend to change the structure and prefer to be as CoDeSys as shown in Example 1 below.

XM-Pro CPU Addressing Existing Scheme				
Block No	**Type**	**Logical Address Range**	**Used as**	**Remark**
0	Output address	Q0:000 to Q0:255	Word and Bit	Physical Digital+Analog Outputs. (local+Expn+Remote)
1	Input address	I1:000 to I1:255	Word and Bit	Physical Digital+Analog Inputs. (local+Expn+Remote)
2	Flags (Memory bits)	F2.000 to F2.255	Bit	Memory bits
3	Status	S3:000 to S3:255	Word	PLC Status & diagnostics
4	Integer Word	W4:000 to W4:255	Word	Memory word address
5	Floating Point	P5:000 to P5:255	Real	Memory Real address
6	Timers	T6:000 to T6:255	Word	Timer word address
7	Counters	C7:000 to C7:255	Word	Counter word address
8	Reserved for future	X8.000 to X8.255	Word	
9	Reserved for future	Y9.000 to Y9.255	Word	
10	Auto memory flag	D10:000 to D10:2048	Bit	Auto generated flags not for user, it will create by utility when user added multiple ladder components , function blocks in parallel with one rung

CPU Addressing Scheme - Example 1				
Block No	**Type**	**Logical Address starting range**	**Used as**	**Remark**
	Digital Output address	%QX0.0	Bit	Physical Digital Outputs. (local+Expn+Remote)
	Digital Input address	%IX0.0	Bit	Physical Digital Inputs. (local+Expn+Remote)
	Analog Output address	%QW0	Int	Physical Analog Outputs. (local+Expn+Remote)
	Analog Input address	%IW0	Int	Physical Analog Inputs. (local+Expn+Remote)
	Output Byte address	%QB0	Byte	
	Input Byte address	%IB0	Byte	
	Flags	%MX0.0	Bit	Memory bits
	WORD/INT	%MW0	WORD	Memory Register address
	DWORD/DINT/Floating Point	%MD0	Dword/Int/Real	Memory Real address
	Status bits	%SX0.0	Bit	PLC Status & diagnostics
	Status Integer/ Word	%SW0	Int/Word	PLC Status & diagnostics
	DWORD/DINT/Floating Point	%DW0	Dword/Int/Real	Memory Real address
	Timers	T0	Timer	Timer word address
	Counters	C0	Counter	Counter word address
	Reserved for future		Word	
	Reserved for future		Word	
	Auto memory flag		Bit	

CPU Addressing Scheme - Example 2

Block No	Type	Logical Address starting range	Used as	Remark
	Digital Output address	Y0	Bit	Physical Digital Outputs. (local+Expn+Remote)
	Digital Input address	X0	Bit	Physical Digital Inputs. (local+Expn+Remote)
	Analog Output address	AY0	Int	Physical Analog Outputs. (local+Expn+Remote)
	Analog Input address	AX0	Int	Physical Analog Inputs. (local+Expn+Remote)
	Flags	M0	Bit	Memory bits
	Byte/WORD/INT	R0	WORD	Memory Register address
	DWORD/DINT/Floating Point	D0	Dword/Int/Real	Memory Real address
	Status bits	SM0	Bit	PLC Status & diagnostics
	Status Integer/ Word	SR0	Int/Word	PLC Status & diagnostics
	Timers	T0	Timer DN bit	Timer word address
	Counters	C0	Counter DN bit	Counter word address
	Reserved for future	K0		

11.Interpreter (Application program)

11.1 Application program interpreter

Logic blocks which is called sequentially in “Main program” tab should create the “MCode” buffer as per the given format.

Eg—

Main Program

Motor1
Motor2
Motor3

Library

Logic blocks

Motor1------50 rungs of logic

Motor2------20 rungs of logic

Motor3------30 rungs of logic

Motor4------60 rungs of logic

In above example under the “Main program” tab we have called the 3 logic blocks. “Motor1”, “Motor2”, “Motor3”. So, 50 + 20 + 30 =100 rungs of MCode should be generated sequentially when user click on compile/generate code option.

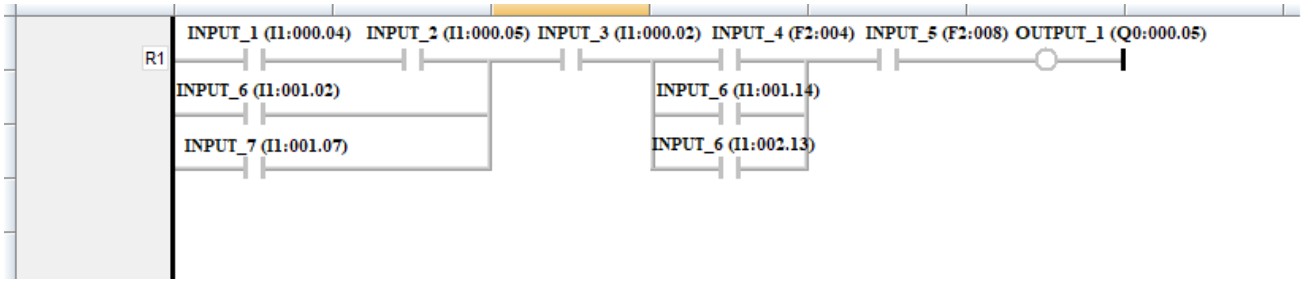
All logic blocks should save in the laptop memory when user opens the existing project all blocks should open as it is.

GUI Logic:

1. When user added any ladder component or function block in a programming grid then at backend the equivalent C code should create.
2. C code should create using the CPU defined address only. Not by user defined tags
3. CPU address details as follows:

XM-Pro CPU Addressing Scheme				
Block No	**Type**	**Logical Address Range**	**Used as**	**Remark**
0	Output address	Q0:000 to Q0:255	Word and Bit	Physical Digital+Analog Outputs. (local+Expn+Remote)
1	Input adress	I1:000 to I1:255	Word and Bit	Physical Digital+Analog Inputs. (local+Expn+Remote)
2	Flags (Memory bits)	F2.000 to F2.255	Bit	Memory bits
3	Status	S3:000 to S3:255	Word	PLC Status & dignostics
4	Integer Word	W4:000 to W4:255	Word	Memory word address
5	Floating Point	P5:000 to P5:255	Real	Memory Real address
6	Timers	T6:000 to T6:255	Word	Timer word address
7	Counters	C7:000 to C7:255	Word	Counter word address
8	Reserved for future	X8.000 to X8.255	Word	
9	Reserved for future	Y9.000 to Y9.255	Word	
10	Auto memory flag	D10:000 to D10:2048	Bit	Auto generated flags not for user, it will create by utility when user added multiple ladder components , function blocks in parallal with one rung

4. Ex-1 : Following is the example for creating Mcode:



Equivalent backend code for above rung-

Q0:000.05=(((I1:000.04 AND I1:000.05) OR I1:001.02 OR I1:001.07) AND (I1:000.02) AND (F2:004 OR I1:001.14 OR I1:002.13) AND F2:008);

5. This equivalent code should convert into Messung “Mcode” format.
6. Messung Mcode format for 1 Rung is:

1	Rung No.	4	byte
2	Data type	4	byte
3	Enable Type	4	byte
4	Enable	4	byte
5	OPCODE	4	byte
6	Type of operand 1	4	byte
7	OP1	4	byte
8	Type of operand 2	4	byte
9	OP2	4	byte
10	Type of operand 3	4	byte
11	OP3	4	byte
12	Type of operand 4	4	byte
13	OP4	4	byte
14	Type of operand 5	4	byte
15	OP5	4	byte
16	Type of operand 6	4	byte
17	OP6	4	byte
18	Type of operand 7	4	byte
19	OP7	4	byte
20	Type of operand 8	4	byte
21	OP8	4	byte
22	No. of Operand	4	byte
23	T_C Name	4	byte
24	Output 1	4	byte
25	Output 2	4	byte

MCODE detailed description is in section 11

7. Above Ex1 equivalent C code conversion example for Mcode buffer creation:

Q0:000.05=(((I1:000.04 AND I1:000.05) OR I1:001.02 OR I1:001.07) AND (I1:000.02) AND (F2:004 OR I1:001.14 OR I1:002.13) AND F2:008);

Rung 1---
D10:000=I1:000.04 AND I1:000.05
Rung 2---
D10:001=D10:000 OR I1:001.02 OR I1:001.07
Rung 3---
D10:002=D10:001 AND I1:000.02
Rung 4---
D10:003=F2:004 OR I1:001.14 OR I1:002.13
Rung 5---
Q0:000.05=D10:002 AND D10:003 AND F2:008

8. Mcode for above Ex1—(not actual calculated address of Oprands just dummy address)

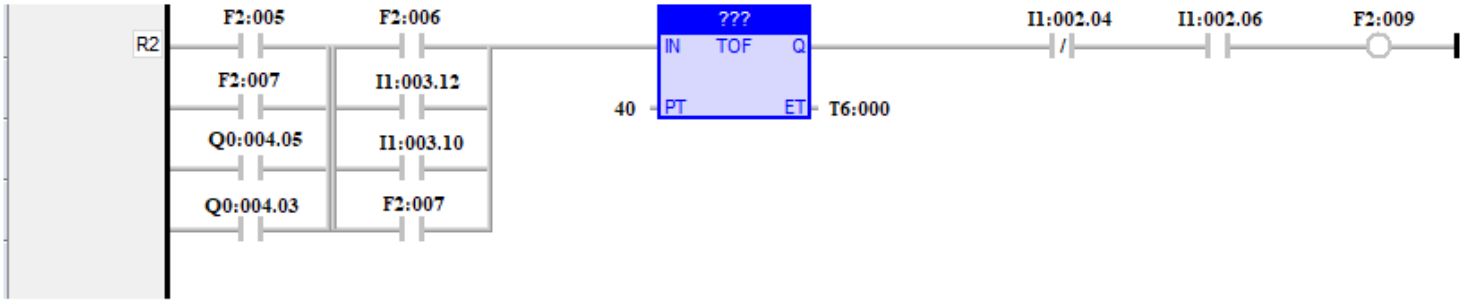
Sr.no.	***Description***	***Actual Mcode Buffer values***	***explanation***
1	Rung No.	0x01	
2	Data type	0x00	
3	Enable Type	0x00	
4	Enable	-	
5	OPCODE	0x0000	AND
6	Type of operand 1	0x01	
7	OP1	0x22204000	Actual address of I1:000.04
8	Type of operand 2	0x01	
9	OP2	0x22204001	Actual address of I1:000.05
10	Type of operand 3	-	
11	OP3	-	
12	Type of operand 4	-	
13	OP4	-	
14	Type of operand 5	-	
15	OP5	-	
16	Type of operand 6	-	
17	OP6	-	
18	Type of operand 7	-	
19	OP7	-	
20	Type of operand 8	-	
21	OP8	-	
22	No. of Operand	0x02	
23	T_C Name	-	
24	Output 1	0x20034445	Actual address of D10:000
25	Output 2	-	
26	Rung No.	0x02	
27	Data type	0x00	

28	Enable Type	0x00	
29	Enable	-	
30	OPCODE	0X0010	OR
31	Type of operand 1	0x01	
32	OP1	0x22204002	Actual address of D10:000
33	Type of operand 2	0x01	
34	OP2	0x22204003	Actual address of I1:001.02
35	Type of operand 3	0x01	
36	OP3	0x22204004	Actual address of I1:001.07
37	Type of operand 4	-	
38	OP4	-	
39	Type of operand 5	-	
40	OP5	-	
41	Type of operand 6	-	
42	OP6	-	
43	Type of operand 7	-	
44	OP7	-	
45	Type of operand 8	-	
46	OP8	-	
47	No. of Operand	0x03	
48	T_C Name	-	
49	Output 1	0x20034447	Actual address of D10:001
50	Output 2	-	
51	Rung No.	0x03	
52	Data type	0x00	
53	Enable Type	0x00	
54	Enable	-	
55	OPCODE	0X0000	AND
56	Type of operand 1	0x01	
57	OP1	0x22204005	D10:001
58	Type of operand 2	0x01	
59	OP2	0x22204006	I1:000.02
60	Type of operand 3	-	
61	OP3	-	
62	Type of operand 4	-	
63	OP4	-	
64	Type of operand 5	-	
65	OP5	-	
66	Type of operand 6	-	
67	OP6	-	
68	Type of operand 7	-	
69	OP7	-	
70	Type of operand 8	-	
71	OP8	-	
72	No. of Operand	0x02	

73	T_C Name	-	
74	Output 1	0x20034447	D10:002
75	Output 2	-	
76	Rung No.	0x04	
77	Data type	0x00	
78	Enable Type	0x00	
79	Enable	-	
80	OPCODE	0X0010	OR
81	Type of operand 1	0x01	
82	OP1	0x22204008	F2:004
83	Type of operand 2	0x01	
84	OP2	0x22204009	I1:001.14
85	Type of operand 3	0x01	
86	OP3	0x22204010	I1:002.13
87	Type of operand 4	-	
88	OP4	-	
89	Type of operand 5	-	
90	OP5	-	
91	Type of operand 6	-	
92	OP6	-	
93	Type of operand 7	-	
94	OP7	-	
95	Type of operand 8	-	
96	OP8	-	
97	No. of Operand	0x03	
98	T_C Name	-	
99	Output 1	0x20034448	D10:003
100	Output 2	-	
101	Rung No.	0x05	
102	Data type	0x00	
103	Enable Type	0x00	
104	Enable	-	
105	OPCODE	0X0000	AND
106	Type of operand 1	0x01	
107	OP1	0x22204011	D10:002
108	Type of operand 2	0x01	
109	OP2	0x22204012	D10:003
110	Type of operand 3	0x01	
111	OP3	0x22204013	F2:008
112	Type of operand 4	-	
113	OP4	-	
114	Type of operand 5	-	
115	OP5	-	
116	Type of operand 6	-	
117	OP6	-	
118	Type of operand 7	-	
119	OP7	-	
120	Type of operand 8	-	

121	OP8	-	
122	No. of Operand	0x03	
123	T_C Name	-	
124	Output 1	0x20034449	Q0:000.05
125	Output 2	-	

9. Ex-2: (Not mentioning user tags just mentioning equivalent address…consider there will be an user tags)



Equivalent C code for above-

F2:009=(TOF0.01S_T151((F2:005 OR F2:007 OR Q0:004.05 OR Q0:004.03) AND (F2:006 OR I1:003.12 OR I1:003.10 OR F2:007)) 40 , T151.Q , T6:000) AND (NOTI1:002.04 AND I1:002.06))

10. Above Ex2 equivalent code conversion example for Mcode buffer creation:

Rung 1---

D10:000=F2:005 OR F2:007 OR Q0:004.05 OR Q0:004.03

Rung 2---

D10:001=F2:006 OR I1:003.12 OR I1:003.10 OR F2:007

Rung 3---

D10:002=D10:000 AND D10:001

Rung 4---

D10:003=NOTI1:002.04 AND I1:002.06

Rung 5---

D10:004,T6:000 =TOF 0.01S_T151(D10:002 , 40)

Rung 6---

F2:009=D10:004 AND D10:003

11. Mcode for above Ex2— (not actual calculated address of Oprands just dummy address)

Sr.no.	*Description*	*Actual Mcode Buffer values*	*Explanation*
1	Rung No.	0x01	
2	Data type	0x00	
3	Enable Type	0x00	
4	Enable	-	
5	OPCODE	0x0010	OR
6	Type of operand 1	0x01	
7	OP1	0x22204000	F2:005
8	Type of operand 2	0x01	
9	OP2	0x22204001	F2:007
10	Type of operand 3	0x01	
11	OP3	0x22204002	Q0:004.05
12	Type of operand 4	0x01	
13	OP4	0x22204003	Q0:004.03
14	Type of operand 5	-	
15	OP5	-	
16	Type of operand 6	-	
17	OP6	-	
18	Type of operand 7	-	
19	OP7	-	
20	Type of operand 8	-	
21	OP8	-	
22	No. of Operand	0x04	
23	T_C Name	-	
24	Output 1	0x20034447	D10:000
25	Output 2	-	
26	Rung No.	0x02	
27	Data type	0x00	
28	Enable Type	0x00	
29	Enable	-	
30	OPCODE	0X0010	OR
31	Type of operand 1	0x01	
32	OP1	0x22204012	F2:006
33	Type of operand 2	0x01	
34	OP2	0x22204013	I1:003.12
35	Type of operand 3	0x01	
36	OP3	0x22204014	I1:003.10
37	Type of operand 4	0x01	
38	OP4	0x22204015	F2:007
39	Type of operand 5	-	
40	OP5	-	
41	Type of operand 6	-	
42	OP6	-	
43	Type of operand 7	-	

44	OP7	-	
45	Type of operand 8	-	
46	OP8	-	
47	No. of Operand	0x04	
48	T_C Name	-	
49	Output 1	0x20034447	D10:001
50	Output 2	-	
51	Rung No.	0x03	
52	Data type	0x00	
53	Enable Type	0x00	
54	Enable	-	
55	OPCODE	0X0000	AND
56	Type of operand 1	0x01	
57	OP1	0x22204015	D10:000
58	Type of operand 2	0x01	
59	OP2	0x22204016	D10:001
60	Type of operand 3	-	
61	OP3	-	
62	Type of operand 4	-	
63	OP4	-	
64	Type of operand 5	-	
65	OP5	-	
66	Type of operand 6	-	
67	OP6	-	
68	Type of operand 7	-	
69	OP7	-	
70	Type of operand 8	-	
71	OP8	-	
72	No. of Operand	0x02	
73	T_C Name	-	
74	Output 1	0x20034447	D10:002
75	Output 2	-	
76	Rung No.	0x04	
77	Data type	0x00	
78	Enable Type	0x00	
79	Enable	-	
80	OPCODE	0X0000	AND
81	Type of operand 1	0x02	
82	OP1	0x22204008	NOTI1:002.04
83	Type of operand 2	0x01	
84	OP2	0x22204009	I1:002.06
85	Type of operand 3	-	
86	OP3	-	I1:002.13
87	Type of operand 4	-	
88	OP4	-	
89	Type of operand 5	-	
90	OP5	-	
91	Type of operand 6	-	

92	OP6	-	
93	Type of operand 7	-	
94	OP7	-	
95	Type of operand 8	-	
96	OP8	-	
97	No. of Operand	0x02	
98	T_C Name	-	
99	Output 1	0x20034448	D10:003
100	Output 2	-	
101	Rung No.	0x05	
102	Data type	**0x07**	
103	Enable Type	0x00	
104	Enable	-	
105	OPCODE	0x01E7	TOF0.01S
106	Type of operand 1	0x01	
107	OP1	0x22204011	D10:002
108	Type of operand 2	0x03	
109	OP2	0x22204012	40
110	Type of operand 3	-	
111	OP3	-	
112	Type of operand 4	-	
113	OP4	-	
114	Type of operand 5	-	
115	OP5	-	
116	Type of operand 6	-	
117	OP6	-	
118	Type of operand 7	-	
119	OP7	-	
120	Type of operand 8	-	
121	OP8	-	
122	No. of Operand	0x02	
123	T_C Name	T151	
124	Output 1	0x20034450	D10:004
125	Output 2	0x20037450	T6:000
126	Rung No.	0x06	
127	Data type	**0x00**	
128	Enable Type	0x00	
129	Enable	-	
130	OPCODE	0x0000	AND
131	Type of operand 1	0x01	
132	OP1	0x22204011	D10:004
133	Type of operand 2	0x01	
134	OP2	0x22204012	D10:003
135	Type of operand 3	-	
136	OP3	-	
137	Type of operand 4	-	
138	OP4	-	
139	Type of operand 5	-	

140	OP5	-	
141	Type of operand 6	-	
142	OP6	-	
143	Type of operand 7	-	
144	OP7	-	
145	Type of operand 8	-	
146	OP8	-	
147	No. of Operand	0x02	
148	T_C Name	-	
149	Output 1	0x20038450	F2:009
150	Output 2	-	

12. Before first rung and after last rung there will be some addition like beow.

Before first Rung-

Description			
SOF	$	1	byte
No. of Rungs	MAX 1000	2	byte

After Last Rung-

Description			
EOF	#	1	byte

12. MCode details.

12.1 SOF- Start of Frame ---

We have defined "$" as a start of Mcode buffer.
Size - 1 byte

12.2 No. of Rungs---

No. of rungs will be calculate after compilation of code of total no. of Mcode rungs.

Size-2 bytes

12.3 Rung no.---

Rung no. ---Rung no as per sequential flow of program
Size- 2 bytes

12.4 Data Type---

As per Opcode –(defined earlier same as per XMPS-100)
Size- 2 bytes

Datatype representation in OPCODE	Description
0x0000	BOOL
0x0001	BYTE
0x0002	WORD
0x0003	DOUBLE WORD
0x0004	INT
0x0005	REAL
0x0006	TON
0x0007	TOFF
0x0008	CTU
0x0009	CTD
0x000A	TP

12.5 Enable type---

This is applicable when FB with enable is used by user.
Size- 1 byte

12.6 Enable—

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.7 Opcode---

This is the Opcode of Rung operation as per below table.
Size- 2 bytes

Instruction representation in OPCODE	Description
0x000x	AND
0x001x	OR
0x002x	XOR
0x003x	NOT
0x004x	ADD
0x005x	SUB

0x006x	MUL
0x007x	DIV
0x008x	MOD
0x009x	MOV
0x00Ax	SHL
0x00Bx	SHR
0x00Cx	ROR
0x00Dx	ROL
0x00Ex	MAX
0x00Fx	MIN
0x010x	LIMIT
0x011x	GT
0x012x	GE
0x013x	LT
0x014x	LE
0x015x	EQ
0X016X	NE
0x017x	Rising Edge
0x018x	Falling Edge
0x019x	CTU
0x01Ax	CTD
0x01Bx	0.01S TON
0x01Cx	0.1TON
0x01Dx	1s TON
0x01Ex	0.01S TOFF
0x01Fx	0.1TOFF
0x020x	1s TOFF
0x021x	0.01s TP
0x022x	0.1s TP
0x023x	1s TP
0x024x	RS
0x025x	SR

12.8 Type of operand 1---

Normal, Negation or Numeric—for logical instructions
Normal, Numeric—for other instructions

(same as XMPS-100)
Size- 1 byte

12.9 OP1---

Oprand 1

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.10 Type of operand 2---

Normal, Negation or Numeric—for logical instructions
Normal, Numeric—for other instructions
(same as XMPS-100)
Size- 1 byte

12.11 OP2---

Oprand 2

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.12 Type of operand 3---

Normal, Negation or Numeric—for logical instructions
Normal, Numeric—for other instructions
(same as XMPS-100)
Size- 1 byte

12.13 OP3---

Oprand 3

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.14 Type of operand 5---

Normal, Negation or Numeric—for logical instructions
Normal, Numeric—for other instructions
(same as XMPS-100)
Size- 1 byte

12.15 OP5---

Oprand 5

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.16 Type of operand 6---

Normal, Negation or Numeric—for logical instructions
Normal, Numeric—for other instructions
(same as XMPS-100)
Size- 1 byte

12.17 OP6---

Oprand 6

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.18 Type of operand 7---

Normal, Negation or Numeric—for logical instructions
Normal, Numeric—for other instructions
(same as XMPS-100)
Size- 1 byte

12.19 OP7---

Oprand 7

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.20 Type of operand 8---

Normal, Negation or Numeric—for logical instructions
Normal, Numeric—for other instructions
(same as XMPS-100)
Size- 1 byte

12.21 OP8---

Oprand 8

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.22 No. of Operand---

We have given max 8 address for any instruction.

User can use greater than 1 address for any instruction.

So for perticuler rung how many operands are used that count should be here.

12.23 T_C Name---

Timer Counter No.

We have limit the Max 255 all types of Timers & Max 255 all types of counters.

So according to this it will increase. And do not repeat.

12.24 Output1---

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

Size- 4 bytes

12.25 Output2---

Here actual MCU address will come….(after conversion of I1,Q0,F2 etc)

During compilation/ generate code command each address will convert into MCU address. (Check section xxx for more details)

This is right now applicable for Time 7 Counter instructions only.

Size- 4 bytes

12.26 EOF---

End of frame—"#"

After generation of Mcode for all backend rungs at last this character should add.

Size-1 byte

13.CPU address to MCU address conversion idea & base address:

Block No	***Type***	***Logical Address Range***	***Total nos PLC address***	***MCU Address starting Address***	***Next address ---***
0	Output address	Q0:000 to Q0:255	256	0X2001 C000	Plus 2
		Q0:000.00...Q0:000.15 to Q0:255.00…Q0:255.15	4096	0x2238 0000	Plus 4
1	Input adress	I1:000 to I1:255	256	0X2001 C200	Plus 2
		I1:000.00…I1:000.15 to I1:255.00…I1:255.15	4096	0X2238 4000	Plus 4
2	Flags (Memory bits)	F2.000 to F2.255	256	0x2238 8000	Plus 4
3	Status	S3:000 to S3:255	256	0x2001 C420	Plus 2
4	Integer Word	W4:000 to W4:255	256	0x2001 C620	Plus 2
5	Floating Point	P5:000 to P5:255	256	0x2001 C820	Plus 4
6	Timers	T6:000 to T6:255	256	0x2001 CC20	Plus 2
7	Counters	C7:000 to C7:255	256	0x2001 CE20	Plus 2
8	Reserved for future	X8.000 to X8.255	256	0x2001 D020	Plus 2
9	Reserved for future	Y9.000 to Y9.255	256	0x2001 D220	Plus 2
10	Auto memory flag	D10:000 to D10:2048	2048	0x223A8400	Plus 4

14. Interpreter (PLC configuration settings)

We need byte by byte data of all PLC configuration settings.

We will download the same using TFTP or TCP same as per the Application Mcode data.

Configuration interpreter will intrerpret the PLC configuration settings in below format:

Sr.no.	*Data Sequence*	*Codes*
1	SOF	#
2	COM Settings	0x31
3	Ethernet Settings	0x32
4	PLC model	0x33
5	PLC onboard IO setting & count	0x34
6	Expansion IO	0x35
7	Remote IO	0x36
8	Modbus RTU requests	0x37
9	Modbus TCP server requests	0x38
10	Modbus TCP client requests	0x39
11	Retentive address	0x3A
12	Initial value of address	0x3B
13	EOF	&

Example:

This is only example not actual address or data.

We need all data in hex format only.

We need only Data coloumn . (Comments are given for understanding of data.)

PLC Configuration Settings

Byte no.	Data	Comment
0	#	SOF
1	0x31	Com settings
2	0 to 4	Baudrate
3	0 to 2	datalength
4	0 to 1	stopbit
5	0 to 3	parity
6	0	send delay
7	0	

8	35	minimum_interface
9	0	
10	0x32	ethernet settings
11	0 /1	use dhcp /static
12	192	IP address 1st byte
13	168	IP address 2nd byte
14	15	IP address 3rd byte
15	15	IP address 4th byte
16	255	Subnet 1st byte
17	255	Subnet 2nd byte
18	255	Subnet 3rd byte
19	0	Subnet 4th byte
20	192	Gateway 1st byte
21	168	Gateway 2nd byte
22	15	Gateway 3rd byte
23	1	Gateway 4th byte
24	502	port no
25	0	
26	0x33	PLC model
27	1 to 2	XM-14DT/XM-17-ADT
28	0x34	PLC on board IO settings
29	0 to 4	AI1 mode
30	0 to 4	AI2 mode
31	0 to 4	AO1 mode
32	10	Total DI count
33	20	Total DO count
34	2	Total AI count
35	2	Total AO count
36	0x35	Expansion IO
37	1 to 5	Total local expansion modules connected
38	xxxx	Model code eg- XM-DI-16 code is 0x12
39	0x22	Logical address assigned CPU address eg.- I1:001.00
40	0x22	
41	0x65	
42	0x87	
43	0x22	Logical address assigned CPU address eg.- I1:001.01
44	0x22	
45	0x65	
46	0x87	
47	0x22	Logical address assigned CPU address eg.- I1:001.02
48	0x22	

49	0x65	
50	0x87	
51	0x22	Logical address assigned CPU address eg.- I1:001.03
52	0x22	
53	0x65	
54	0x87	
55	0x22	Logical address assigned CPU address eg.- I1:001.04
56	0x22	
57	0x65	
58	0x87	
59	0x22	Logical address assigned CPU address eg.- I1:001.05
60	0x22	
61	0x65	
62	0x87	
63	0x22	Logical address assigned CPU address eg.- I1:001.06
64	0x22	
65	0x65	
66	0x87	
67	0x22	Logical address assigned CPU address eg.- I1:001.07
68	0x22	
69	0x65	
70	0x87	
71	0x22	Logical address assigned CPU address eg.- I1:001.08
72	0x22	
73	0x65	
74	0x87	
75	0x22	Logical address assigned CPU address eg.- I1:001.09
76	0x22	
77	0x65	
78	0x87	
79	0x22	Logical address assigned CPU address eg.- I1:001.10
80	0x22	
81	0x65	
82	0x87	
83	0x22	Logical address assigned CPU address eg.- I1:001.11
84	0x22	
85	0x65	
86	0x87	
87	0x22	Logical address assigned CPU address eg.- I1:001.12
88	0x22	
89	0x65	

90	0x87	
91	0x22	Logical address assigned CPU address eg.- I1:001.13
92	0x22	
93	0x65	
94	0x87	
95	0x22	Logical address assigned CPU address eg.- I1:001.14
96	0x22	
97	0x65	
98	0x87	
99	0x22	Logical address assigned CPU address eg.- I1:001.15
100	0x22	
101	0x65	
102	0x87	
103	0x35	Expansion IO (3 for
104	xxxx	Model code eg- XM-DI-16 code is 0x12
105	0x22	Logical address assigned CPU address eg.- I1:002.00
106	0x22	
107	0x65	
108	0x87	
109	0x22	Logical address assigned CPU address eg.- I1:002.01
110	0x22	
111	0x65	
112	0x87	
113	0x22	Logical address assigned CPU address eg.- I1:002.02
114	0x22	
115	0x65	
116	0x87	
117	0x22	Logical address assigned CPU address eg.- I1:002.03
118	0x22	
119	0x65	
120	0x87	
121	0x22	Logical address assigned CPU address eg.- I1:002.04
122	0x22	
123	0x65	
124	0x87	
125	0x22	Logical address assigned CPU address eg.- I1:002.05
126	0x22	
127	0x65	
128	0x87	
129	0x22	Logical address assigned CPU address eg.- I1:002.06
130	0x22	

131	0x65	
132	0x87	
133	0x22	Logical address assigned CPU address eg.- I1:002.07
134	0x22	
135	0x65	
136	0x87	
137	0x22	Logical address assigned CPU address eg.- I1:002.08
138	0x22	
139	0x65	
140	0x87	
141	0x22	Logical address assigned CPU address eg.- I1:002.09
142	0x22	
143	0x65	
144	0x87	
145	0x22	Logical address assigned CPU address eg.- I1:002.10
146	0x22	
147	0x65	
148	0x87	
149	0x22	Logical address assigned CPU address eg.- I1:002.11
150	0x22	
151	0x65	
152	0x87	
153	0x22	Logical address assigned CPU address eg.- I1:002.12
154	0x22	
155	0x65	
156	0x87	
157	0x22	Logical address assigned CPU address eg.- I1:002.13
158	0x22	
159	0x65	
160	0x87	
161	0x22	Logical address assigned CPU address eg.- I1:002.14
162	0x22	
163	0x65	
164	0x87	
165	0x22	Logical address assigned CPU address eg.- I1:002.15
166	0x22	
167	0x65	
168	0x87	
169	0x35	Expansion IO (3 for
170	xxxx	Model code eg- XM-DO-16T code is 0x13
171	0x22	Logical address assigned CPU address eg.- Q0:00Q.00

172	0x22	
173	0x65	
174	0x87	
175	0x22	Logical address assigned CPU address eg.- Q0:00Q.01
176	0x22	
177	0x65	
178	0x87	
179	0x22	Logical address assigned CPU address eg.- Q0:00Q.02
180	0x22	
181	0x65	
182	0x87	
183	0x22	Logical address assigned CPU address eg.- Q0:00Q.03
184	0x22	
185	0x65	
186	0x87	
187	0x22	Logical address assigned CPU address eg.- Q0:00Q.04
188	0x22	
189	0x65	
190	0x87	
191	0x22	Logical address assigned CPU address eg.- Q0:00Q.05
192	0x22	
193	0x65	
194	0x87	
195	0x22	Logical address assigned CPU address eg.- Q0:00Q.06
196	0x22	
197	0x65	
198	0x87	
199	0x22	Logical address assigned CPU address eg.- Q0:00Q.07
200	0x22	
201	0x65	
202	0x87	
203	0x22	Logical address assigned CPU address eg.- Q0:00Q.08
204	0x22	
205	0x65	
206	0x87	
207	0x22	Logical address assigned CPU address eg.- Q0:00Q.09
208	0x22	
209	0x65	
210	0x87	
211	0x22	Logical address assigned CPU address eg.- Q0:00Q.10
212	0x22	

213	0x65	
214	0x87	
215	0x22	Logical address assigned CPU address eg.- Q0:00Q.11
216	0x22	
217	0x65	
218	0x87	
219	0x22	Logical address assigned CPU address eg.- Q0:00Q.12
220	0x22	
221	0x65	
222	0x87	
223	0x22	Logical address assigned CPU address eg.- Q0:00Q.13
224	0x22	
225	0x65	
226	0x87	
227	0x22	Logical address assigned CPU address eg.- Q0:00Q.14
228	0x22	
229	0x65	
230	0x87	
231	0x22	Logical address assigned CPU address eg.- Q0:00Q.15
232	0x22	
233	0x65	
234	0x87	
235	0x35	Expansion IO
236	xxxx	Model code eg- XM-AI2-AO2 code is 0x14
237	0 to 4	Mode of AI1
238	0 to 4	Mode of AI2
239	0 to 4	Mode of AO1
240	0 to 4	Mode of AO2
241	0x22	Logical address assigned CPU address eg.- AI1-I0:003
242	0x22	
243	0x65	
244	0x87	
245	0x22	Logical address assigned CPU address eg.- AI2-I0:004
246	0x22	
247	0x65	
248	0x87	
249	0x22	
250	0x22	Logical address assigned CPU address eg.- AO1-Q0:003
251	0x22	
252	0x65	
253	0x87	

254	0x22	Logical address assigned CPU address eg.- AO2-Q0:004
255	0x22	
256	0x65	
257	0x87	
258	0x36	Remote IO
259	1 to 5	Total remote IO modules connected
260	xxxx	Model code eg- MOD-DO-16R code is 0x21
261	0x22	Logical address assigned CPU address eg.- Q0:004.00
262	0x22	
263	0x65	
264	0x87	
265	0x22	Logical address assigned CPU address eg.- Q0:004.01
266	0x22	
267	0x65	
268	0x87	
269	0x22	Logical address assigned CPU address eg.- Q0:004.02
270	0x22	
271	0x65	
272	0x87	
273	0x22	Logical address assigned CPU address eg.- Q0:004.03
274	0x22	
275	0x65	
276	0x87	
277	0x22	Logical address assigned CPU address eg.- Q0:004.04
278	0x22	
279	0x65	
280	0x87	
281	0x22	Logical address assigned CPU address eg.- Q0:004.05
282	0x22	
283	0x65	
284	0x87	
285	0x22	Logical address assigned CPU address eg.- Q0:004.06
286	0x22	
287	0x65	
288	0x87	
289	0x22	Logical address assigned CPU address eg.- Q0:004.07
290	0x22	
291	0x65	
292	0x87	
293	0x22	Logical address assigned CPU address eg.- Q0:004.08
294	0x22	

295	0x65	
296	0x87	
297	0x22	Logical address assigned CPU address eg.- Q0:004.09
298	0x22	
299	0x65	
300	0x87	
301	0x22	Logical address assigned CPU address eg.- Q0:004.10
302	0x22	
303	0x65	
304	0x87	
305	0x22	Logical address assigned CPU address eg.- Q0:004.11
306	0x22	
307	0x65	
308	0x87	
309	0x22	Logical address assigned CPU address eg.- Q0:004.12
310	0x22	
311	0x65	
312	0x87	
313	0x22	Logical address assigned CPU address eg.- Q0:004.13
314	0x22	
315	0x65	
316	0x87	
317	0x22	Logical address assigned CPU address eg.- Q0:004.14
318	0x22	
319	0x65	
320	0x87	
321	0x22	Logical address assigned CPU address eg.- Q0:004.15
322	0x22	
323	0x65	
324	0x87	
325	0x36	Remote IO
326	xxxx	Model code eg- MOD-DI-8 code is 0x22
327	0x22	Logical address assigned CPU address eg.- I1:005.00
328	0x22	
329	0x65	
330	0x87	
331	0x22	Logical address assigned CPU address eg.- I1:005.01
332	0x22	
333	0x65	
334	0x87	
335	0x22	Logical address assigned CPU address eg.- I1:005.02

336	0x22	
337	0x65	
338	0x87	
339	0x22	Logical address assigned CPU address eg.- I1:005.03
340	0x22	
341	0x65	
342	0x87	
343	0x22	Logical address assigned CPU address eg.- I1:005.04
344	0x22	
345	0x65	
346	0x87	
347	0x22	Logical address assigned CPU address eg.- I1:005.05
348	0x22	
349	0x65	
350	0x87	
351	0x22	Logical address assigned CPU address eg.- I1:005.06
352	0x22	
353	0x65	
354	0x87	
355	0x22	Logical address assigned CPU address eg.- I1:005.07
356	0x22	
357	0x65	
358	0x87	
359	0x22	Logical address assigned CPU address eg.- I1:005.08
360	0x22	
361	0x65	
362	0x87	
363	0x22	Logical address assigned CPU address eg.- I1:005.09
364	0x22	
365	0x65	
366	0x87	
367	0x22	Logical address assigned CPU address eg.- I1:005.10
368	0x22	
369	0x65	
370	0x87	
371	0x22	Logical address assigned CPU address eg.- I1:005.11
372	0x22	
373	0x65	
374	0x87	
375	0x22	Logical address assigned CPU address eg.- I1:005.12
376	0x22	

377	0x65	
378	0x87	
379	0x22	Logical address assigned CPU address eg.- I1:005.13
380	0x22	
381	0x65	
382	0x87	
383	0x22	Logical address assigned CPU address eg.- I1:005.14
384	0x22	
385	0x65	
386	0x87	
387	0x22	Logical address assigned CPU address eg.- I1:005.15
388	0x22	
389	0x65	
390	0x87	
391	0x36	Remote IO
392	xxxx	Model code eg- MOD-AI2-AO2 code is 0x24
393	0 to 4	Mode of AI1
394	0 to 4	Mode of AI2
395	0 to 4	Mode of AO1
396	0 to 4	Mode of AO2
397	0x22	Logical address assigned CPU address eg.- AI1-I0:006
398	0x22	
399	0x65	
400	0x87	
401	0x22	Logical address assigned CPU address eg.- AI1-I0:007
402	0x22	
403	0x65	
404	0x87	
405	0x22	Logical address assigned CPU address eg.- AO1-Q0:006
406	0x22	
407	0x65	
408	0x87	
409	0x22	Logical address assigned CPU address eg.- AO1-Q0:007
410	0x22	
411	0x65	
412	0x87	
413	0x37	Modbus RTU requests
414	3	Total no of requests
415	1	Slave id
416	0b	com.timeout
417	b8	

418	2	no. of retries
419	0	polling
420	10	
421	0x22	variable CPU address
422	0x23	
423	0x24	
424	0x25	
425	0x00	data start address
426	0x01	
427	0x00	data size
428	0x01	
429	0x06	function code
430	2	Slave id
431	0b	com.timeout
432	b9	
433	12	no. of retries
434	16	polling
435	20	
436	0x22	variable CPU address
437	0x23	
438	0x24	
439	0x25	
440	0x00	data start address
441	0x01	
442	0x00	data size
443	0x01	
444	0x06	function code
445	3	Slave id
446	0b	com.timeout
447	b9	
448	12	no. of retries
449	16	polling
450	20	
451	0x22	variable CPU address
452	0x23	
453	0x24	
454	0x25	
455	0x00	data start address
456	0x01	
457	0x00	data size
458	0x01	

459	0x06	function code
460	4	Slave id
461	0b	com.timeout
462	b10	
463	24	no. of retries
464	28	polling
465	32	
466	0x22	variable CPU address
467	0x23	
468	0x24	
469	0x25	
470	0x00	data start address
471	0x01	
472	0x00	data size
473	0x01	
474	0x06	function code
475	0x38	Modbus TCP server requests
476	1	Total no of requests
477	0x00	Port
478	0x02	
479	0x22	variable CPU address
480	0x23	
481	0x24	
482	0x25	
483	0x00	data start address
484	0x01	
485	0x00	length
486	0x01	
487	0x06	function code
488	0x39	Modbus TCP client requests
489	1	Total no of requests
490	192	Slave address
491	168	
492	15	
493	30	
494	0x00	Port
495	0x02	
496	0	polling
497	10	
498	0x01	device id
499	0x22	variable CPU address

500	0x23	
501	0x24	
502	0x25	
503	0x00	data start address
504	0x01	
505	0x00	length
506	0x01	
507	0x06	function code
508	0x3A	Retentive address
509	1	No of retentive address
510	0x22	Main address (CPU convertible address value)
511	0x23	
512	0x24	
513	0x25	
514	0x40	Retentive address (CPU convertible address value)
515	0x41	
516	0x42	
517	0x43	
518	0x3B	Initial value of address
519	1	Total No of address
520	0x22	Main address (CPU convertible address value)
521	0x23	
522	0x24	
523	0x25	
524	0x67	Data
525	0x00	
526	0x67	
527	0x00	
528	&	EOF

Data should be hex format
This is example data for understanding only.