SYNTEC TECHNOLOGY CO.,LTD.

PLC Editor Operation Manual.

匯出日期：2021-08-03
修改日期：2019-12-19

中文文件 Mandarin Document: OpenCNC_PLC发展工具操作手册

1 Ladder Programming and Development.

1.1 M/S/T Code

1.1.1 Functions and Purposes of M/S/T Code

NC programs normally use M code to drive the auxiliary devices like oil cooler or lubricant; use S code to set up spindle speed; use T code to change tool.

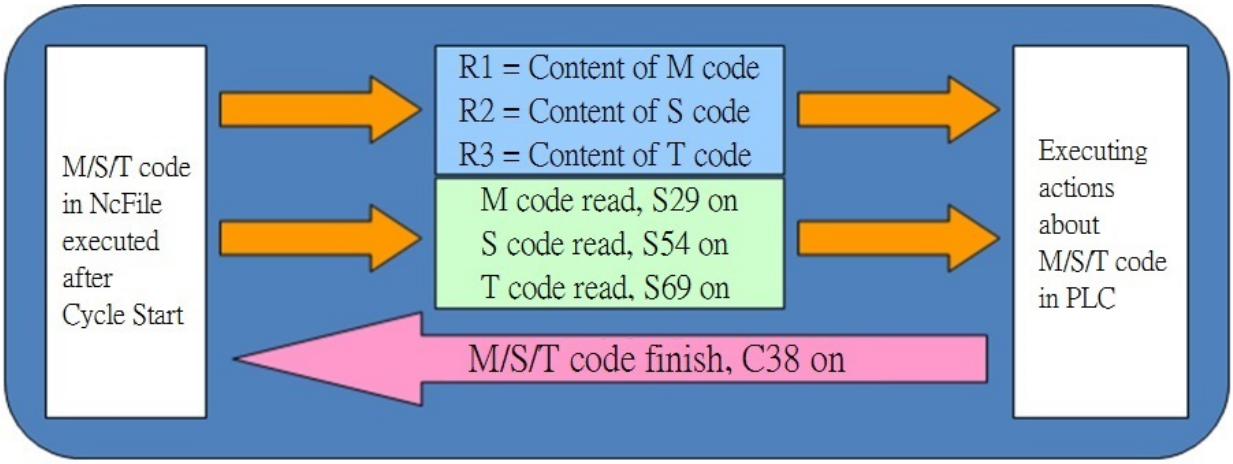
By applying M/S/T code, it'll be able to execute synchronous control of external devices in NC program through the ladder diagram.

NC Program	Action Instruction
N1 G00 X50.0 Y100.0 N2 T1 N3 S1000 M3 N4 G01 Z-10.0 F3000 N5 M8 N6 G01 Z-15. F500	N1 X axis/Y axis move to 50./100.mm with G00 N2 Change to tool No.1 N3 Spindle rotates at 1000 RPM CW N4 Z axis moves to -10.mm with G01 N5 Open cutting fluid N6 Z axis moves to -15.mm with G01

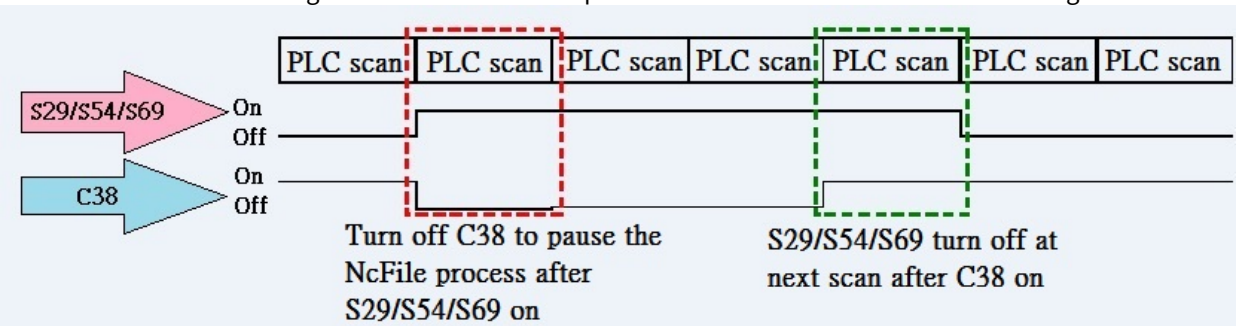
In the example above, the tool changing is executed with T1 in second line of the program; the spindle speed is set to 1000 RPM with S1000 and the CW rotation is started with M3 in the third line; the cutting fluid is opened with M8 in the fifth line.

As for how the controllers deal with M/S/T code and drive the auxiliary devices correctly, there will be an introduction to the structure in following chapters.

1.1.2 Process of M/S/T Code Actions



- When the kernel of the controller executes the M/S/T code in NC programs, contents of the M/S/T code will be saved in R1/R2/R3 registers.
    - For example, when executes M3, the kernel saves 3 into R1, which means R1=3; when executes T5, the kernel saves 5 into R3, which means R3=5.
- When the kernel of the controller executes the M/S/T code, it'll send M/S/T code reading flags to ladder diagram, M/S/T code correspond to S29/S54/S69 respectively.
    - For example, when executes M code, S29 will be triggered and can be used as the start signal of some specific actions.
- The ladder diagram starts executing the organized actions right after received the M/S/T code reading flags.
- If the M/S/T code are written in the same block of the NC program, the kernel will send S29/S54/S69 to the ladder diagram at the same time when executing that block.
- After the organized actions of M/S/T codes are all done, C38 needs to be triggered in ladder diagram as the M/S/T code finish flag.
- The kernel will continue to execute the NC program after receiving C38 signal.
- When editing the ladder diagram, please pay attention to the timing of C38 ON/OFF. The relation between C38 and S29/S54/S69 is shown below.
    - The action in red box is the rule of editing ladder diagram.
    - The action in green box is the natural specification after the kernel detected C38 signal.



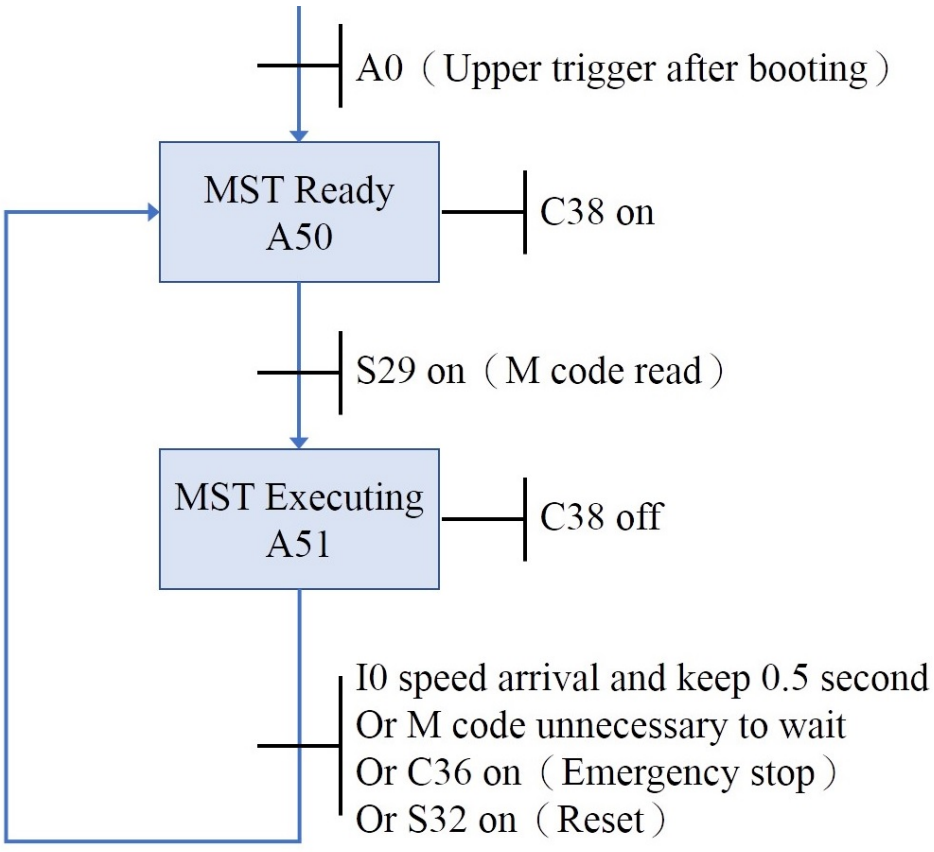
1.1.3 How to write M/S/T code

Basic Rules

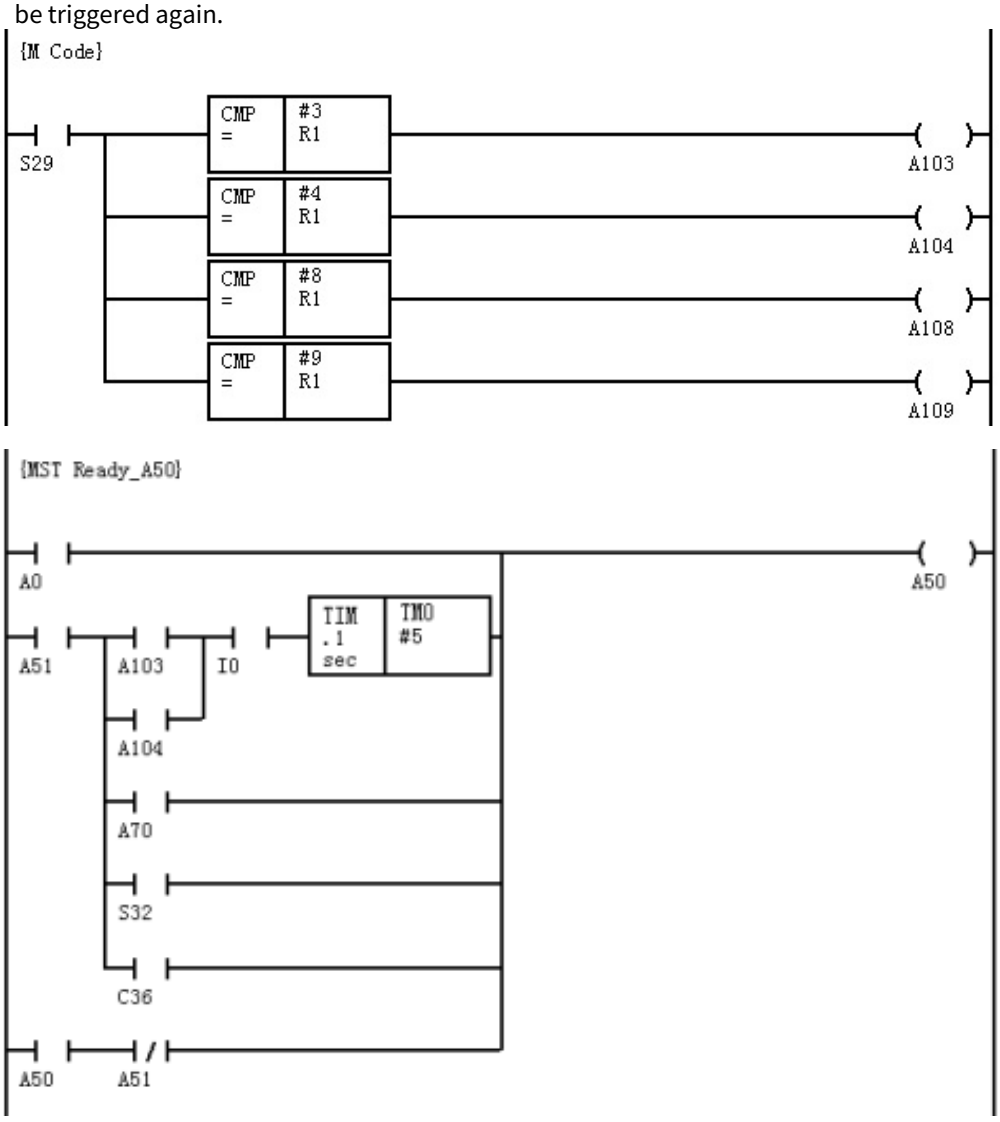
- For M code needs completion confirmation, please write the ladder diagram with the concept of "Machine State Diagram（MSD）".
- In normal state or when not running M/S/T code, it suggests C38 to stay triggered.
- As for M code that don't need completion confirmation such as cutting fluid, working LED, chip conveying device, no need to turn off C38 while executing M code.
- As for M code that need completion confirmation such as spindle CW/CCW rotation, workpiece gripping, spindle orientation, please turn off C38 while executing M code and activate it again after M code is finish.

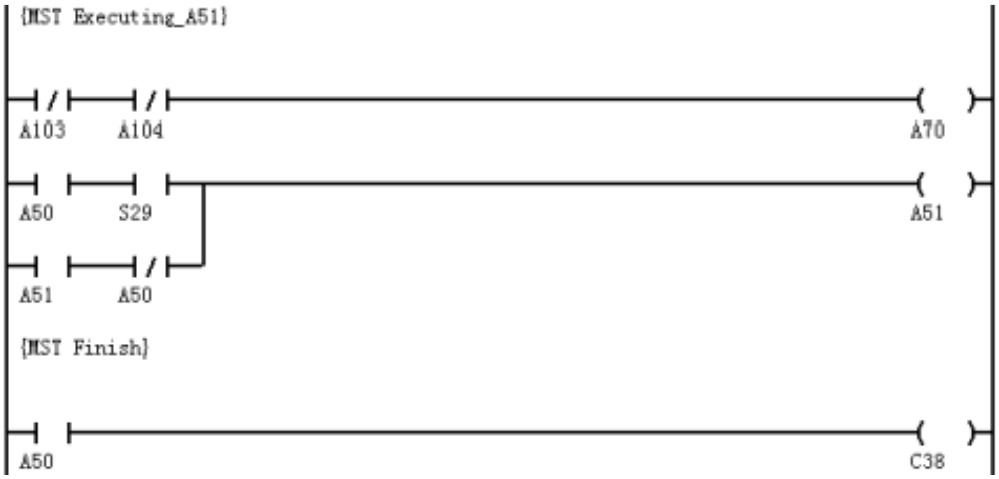
MSD Example

- The picture below separates the execution of M/S/T code with 2 states.
  - When M/S/T code are in ready state（A50）, C38 is conducted.
  - When M/S/T code are in execution state（A51）, C38 is off to make the controller wait.



- **M code that don't need completion confirmation**：Take M8/M9 cutting fluid on/off as example, organize 2 contacts A108/A109 to directly trigger the completion condition of M code, A70, and the controller doesn't need to wait.
- **M code that need completion confirmation**：Take M3/M4 spindle CW/CCW rotation as example, organize 2 contacts A103/A104. Since the spindle rotation is influenced by the acceleration time of the motor, the machining process can only be proceed after the spindle reaches target speed. Thus, in M3/M4 execution state, C38 needs to be off till the speed reaching signal I0 is activated for 0.5 sec, then can be triggered again.





Component List：

I0	Spindle speed reaching	A103	M3 Spindle CW rotation
A50	MST ready	A104	M4 Spindle CCW rotation
A51	MST executing	A108	M8 Cutting fluid ON
A70	No need to wait for M code	A109	M9 Cutting fluid OFF

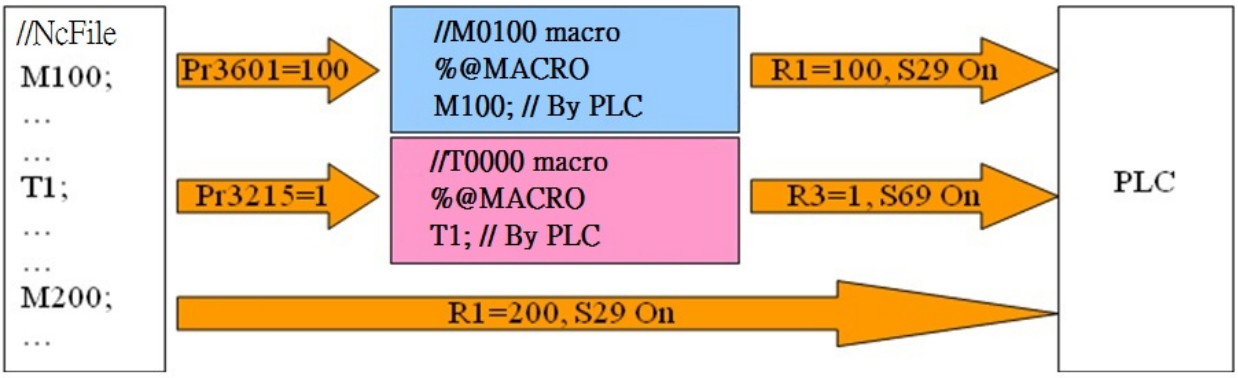
1.1.4 M/T Code Macro Calling Function

SYNTEC controller provides 10 sets of M code macro calling function and 1 set of T code macro calling function.

So M/T code can also use the full macro function besides organizing actions in ladder diagram.

When controller executes the macro calling M/T code, it'll first execute the corresponding macro content and won't trigger S29/S69 or set values into R1/R3.

But S29/S69 or R1/R3 will still be executed if M/T code are written in the macro.



M code macro calling function setup

- Register the macro calling M code in controller parameter Pr3601~3610.
- Prepare the calling macro first and save it in the path below：
  - DOS system：C:CNC/MACRO
  - CE system：DiskC/OpenCnc/MACRO
- The file name of the macro should correspond to the macro calling M code. For example, if Pr3601=50, the macro file name should be M0050（no extension）.

T code macro calling function setup

- Set Pr3215 as you need, reference：Pr3215 *Enable T code call mode.
- Prepare the T code calling macro first and save it in the path below：
  - DOS system：C:CNC/MACRO
  - CE system：DiskC/OpenCnc/MACRO
- The file name of T code calling macro is always T0000（no extension）.

1.1.5 M/S/T Code Additional Notifications

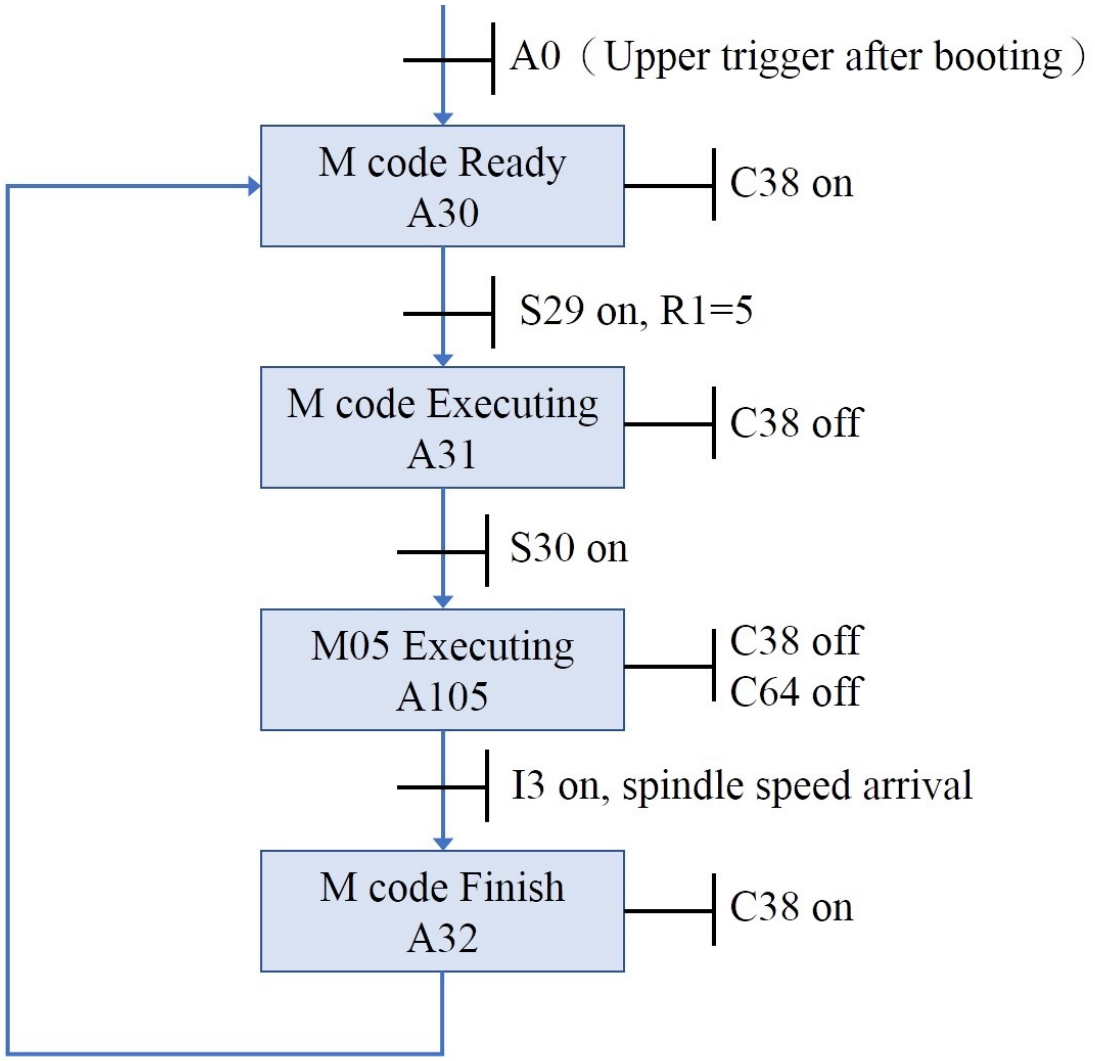
G/M code in the same block

When G code and M code appear in the same block, the controller sends G code action command and M code reading flag（S29） at the same time.

With different properties, some M code can be executed with G code simultaneously（For example：M8 Cutting fluid On）; some others can only be executed after G code is completed（For example: M5 Spindle stop）.

When dealing with the latter situation, it's suggested to use S30（G code completion signal in same block） to add G code completion waiting state in MSD for protection.

Example：When G01 and M5 are in the same block, G01 must be completed first before executing M5.



Multiple M code executing function

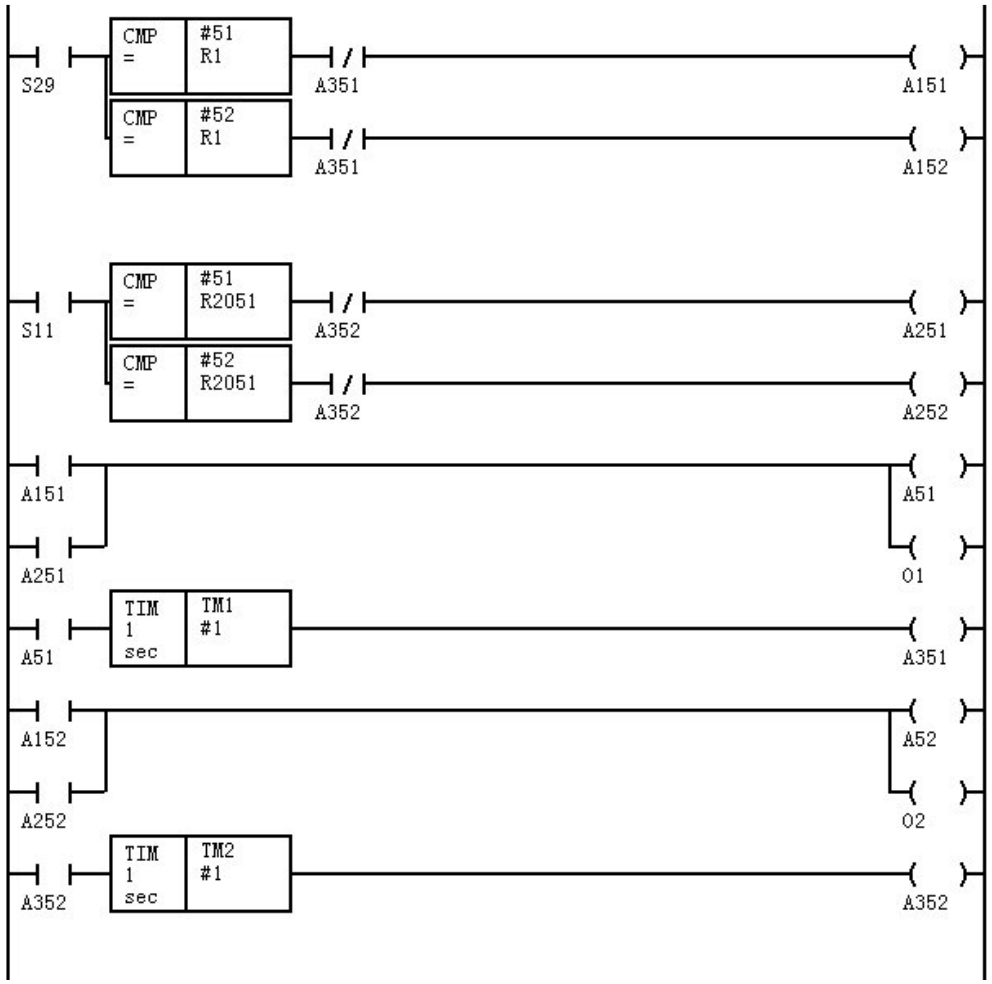
In some applications, users need to execute multiple M code at the same time instead of executing one by one. For the situations, it's suggested to turn on the multiple M code executing function to satisfy the requirement.

- Please refer to the parameter manual: Pr3810 *Paralleled executing multiple M code in one block.
- Table below shows the related PLC components of 4 M/S/T channels after activating Pr3810.

1st Channel	M code read	S29	S11	S12	S13	S14
	M code content	R2050（R1）	R2051	R2052	R2053	R2054
	M code finish	C38				

2nd Channel	M code read	R615.24（R615.2）	R615.25	R615.26	R615.27	R615.28
	M code content	R2060（R616）	R2061	R2062	R2063	R2064
	M code finish	R615.0				
3rd Channel	M code read	R619.24（R619.2）	R619.25	R619.26	R619.27	R619.28
	M code content	R2070（R620）	R2071	R2072	R2073	R2074
	M code finish	R619.0				
4th Channel	M code read	R623.24（R623.2）	R623.25	R623.26	R623.27	R623.28
	M code content	R2080（R624）	R2081	R2082	R2083	R2084
	M code finish	R623.0				1

- Example：The method below is able to output O1 and O2 at the same time no matter the order of M code is M51 M52 or M52 M51.



1.2 How to send alarms with ladder diagram

The alarms sent by ladder diagram will be recorded into the history alarms of the controller automatically for further analysis.

When alarms happened, the controller will only show the alarm on the screen and won't stop any executing actions.

Users should design the actions in the ladder diagram separately if any process needs to be suspended or stopped.

1.2.1 System Built-in Alarms

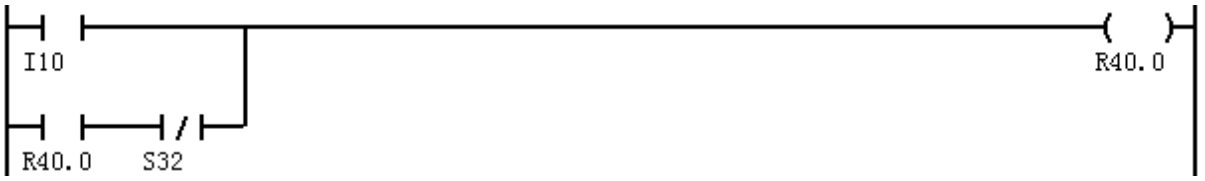
When R40 ~ R43 are outputted in the ladder diagram, alarm windows pop-up on the controller screen according to the table below.

Alarm Number	Output	Alarm Message
001	R40.0	X+ hardware stroked limit exceed
002	R40.1	X- hardware stroked limit exceed
003	R40.2	X not yet back home
004	R40.3	X servo driver alarm
009	R40.8	Y+ hardware stroked limit exceed
010	R40.9	Y- hardware stroked limit exceed
011	R40.10	Y not yet back home
012	R40.11	Y servo driver alarm
017	R41.0	Z+ hardware stroked limit exceed
018	R41.1	Z- hardware stroked limit exceed
019	R41.2	Z not yet back home
020	R41.3	Z servo driver alarm
025	R41.8	A+ hardware stroked limit exceed
026	R41.9	A- hardware stroked limit exceed
027	R41.10	A not yet back home
028	R41.11	A servo driver alarm

033	R42.0	B+ hardware stroked limit exceed
034	R42.1	B- hardware stroked limit exceed
035	R42.2	B not yet back home
036	R42.3	B servo driver alarm
041	R42.8	C+ hardware stroked limit exceed
042	R42.9	C- hardware stroked limit exceed
043	R42.10	C not yet back home
044	R42.11	C servo driver alarm
049	R43.0	Spindle driver alarm
050	R43.1	Coolant motor overload
051	R43.2	Oil press system alarm
052	R43.3	Air pressure too low
053	R43.4	Lube too little
054	R43.5	Coolant level too low
055	R43.6	Clamper not close
056	R43.7	Tool exchange system alarm
057	R43.8	CPU temperature over than 100 degree

Alarm triggering

As the picture below, the external signal I-10 triggers R40.0, and R40.0 corresponds to PLC Alarm No.01.



1.2.2 User-defined Alarms

SYNTEC controller provides 96 PLC Alarms, 96 PLC Warnings and 319 PLC Hints for users to define, the Alarm/ Warning/Hint number and corresponding R bit is shown in the table below.

PLC Alarm number 01~64 are for system built-in alarms, please do not reuse.

Type	R Bit Number	Alarm Number
PLC Alarm	R44.0 ~ R44.15	65 ~ 80
	R45.0 ~ R45.15	81 ~ 96
	R46.0 ~ R46.15	97 ~ 112
	R47.0 ~ R47.15	113 ~ 128
	R48.0 ~ R48.15	129 ~ 144
	R49.0 ~ R49.15	145 ~ 160
PLC Warning	R44.16 ~ R44.31	315 ~ 330
	R45.16 ~ R45.31	331 ~346
	R46.16 ~ R46.31	347 ~ 362
	R47.16 ~ R47.31	363 ~ 378
	R48.16 ~ R48.31	379 ~ 394
	R49.16 ~ R49.31	395 ~ 410
PLC Hint	R590.1 ~ R590.31	1 ~ 31
	R591.0 ~ R591.31	32 ~ 63
	R592.0 ~ R592.31	64 ~ 95
	R593.0 ~ R593.31	96 ~ 127

	R594.0 ~ R594.31	128 ~ 159
	R595.0 ~ R595.31	160 ~ 191
	R596.0 ~ R596.31	192 ~ 223
	R597.0 ~ R597.31	224 ~ 255
	R598.0 ~ R598.31	256 ~ 287
	R599.0 ~ R599.31	288 ~ 319
Note	1. PLC Alarms will be saved in history alarm, but PLC Warning and PLC Hint won't. 2. PLC Hint doesn't need self-hold, the alarm window can be shown on the screen continuously. 3. PLC Alarm and PLC Warning need self-hold in the ladder diagram to make the alarm window stay on the screen.	

DOS System Alarm String File

- File Path

  Traditional/Simplified Chinese：C:CNC/APP/APPCHI.STR
  English：C:CNC/APP/APPENG.STR
  Others：C:CNC/APP/CNCLOC.STR

- File Format

  20xxx="1;MSG=Alarm Content", xxx corresponds to PLC Alarm/Warning number, please note that the identification code is 20。

  23xxx="1;MSG=Alarm Content", xxx corresponds to PLC Hint number, please note that the identification number is 23。

- APPCHI.STR example：

  20001="1;MSG=X轴超过正向硬体行程极限"

  23010="1;MSG=第10号提示"

- APPENG.STR example：

  20001="1;MSG=X+ hardware stroked limit exceed"

  23010="1;MSG=10th Hint"

CE System Alarm String File

- File Path（before version 10.114.46）

Chinese：DiskC/OCRes/CHT/String/AlarmMsg_CHT.xml

English：DiskC/OCRes/Common/String/AlarmMsg_common.xml

- File Path（after version 10.114.46）

  Traditional Chinese：DiskC/OCRes/CHT/String/AlarmPLC_CHT.xml

  Simplified Chinese：DiskC/OCRes/CHS/String/AlarmPLC_CHS.xml

  English：DiskC/OCRes/Common/String/AlarmPLC_COM.xml

- File Format

  <Message ID="AlarmMsg::MLC::ID=xxx" Content="Alarm Content" />, xxx corresponds to PLC Alarm/ Warning number, please note that the identification string is MLC。

  <Message ID="AlarmMsg::MLCHint::ID=xxx" Content="Alarm Content" />, xxx corresponds to PLC Hint number, please note that the identification string is MLCHint。

  The string length of the alarm content is 48 English characters or 31 Chinese words, the rest of the string will exceed the alarm window and won't be shown.

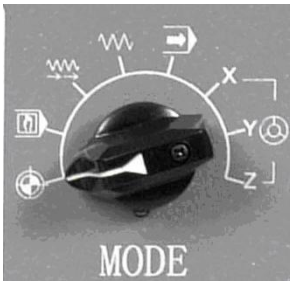
- Example
  - AlarmPLC_CHT.xml：
    - <Message ID="AlarmMsg::MLC::ID=1" Content="X轴超过正向硬体行程极限" />
    - <Message ID="AlarmMsg::MLCHint::ID=10" Content="第10号提示" />
  - AlarmPLC_COM.xml：
    - <Message ID="AlarmMsg::MLC::ID=1" Content="X+ hardware stroked limit exceed " />
    - <Message ID="AlarmMsg::MLCHint::ID=10" Content="10th Hint " />

1.3 Operation Panel Application Example

Mode Selection | Spindle Override | MPG/ INJOG/ G00 Override | G01/ JOG Override | MPG Simulation | Program Dry Run | Optional Stop | Optional Skip | Single Block | Emergency Stop | JOG | Cycle Start/Feed Hold | Home Searching | Hardware Stroke Limit | Spindle CW/CCW/Stop

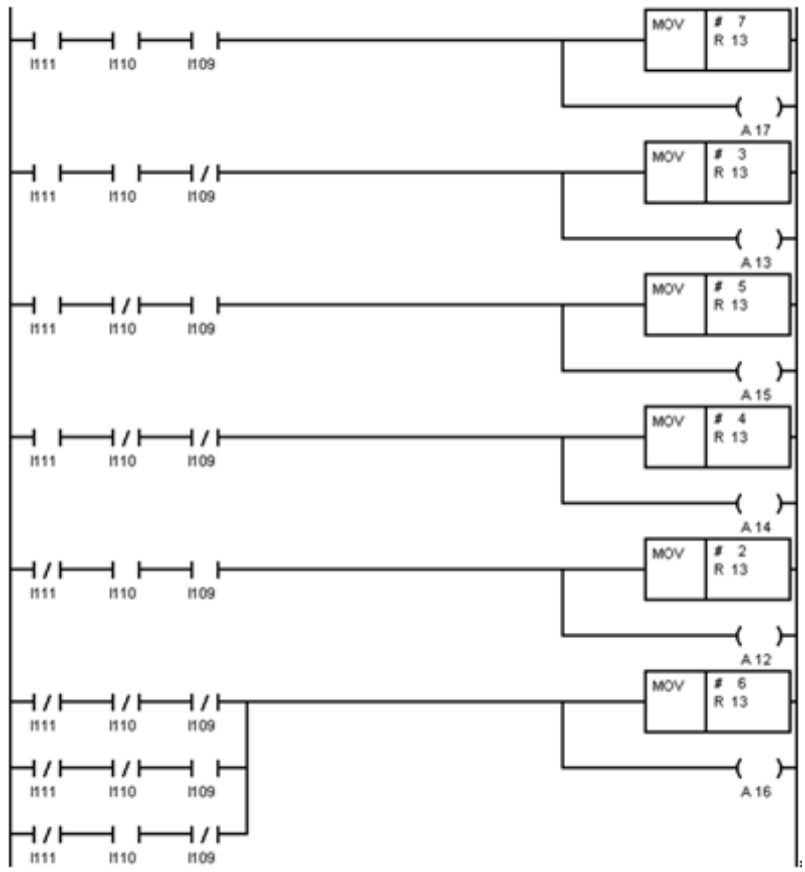
Controller Mode（R13） is composed of three I bits, I-109/ I-110/ I-111, in binary form.

The input/output transfer relationship are shown in the table below.



Input			Output
I-111	I-110	I-109	R13
1	1	1	7：HOME
1	1	0	3：MDI
1	0	1	5：INJOG
1	0	0	4：JOG
0	1	1	2：Auto
0	1	0	6：MPG
0	0	1	6：MPG
0	0	0	6：MPG

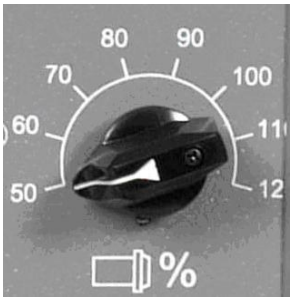
Ladder Diagram Example：



Mode Selection | Spindle Override | MPG/ INJOG/ G00 Override | G01/ JOG Override | MPG Simulation
Program Dry Run | Optional Stop | Optional Skip | Single Block | Emergency Stop | JOG
Cycle Start/Feed Hold | Home Searching | Hardware Stroke Limit | Spindle CW/CCW/Stop

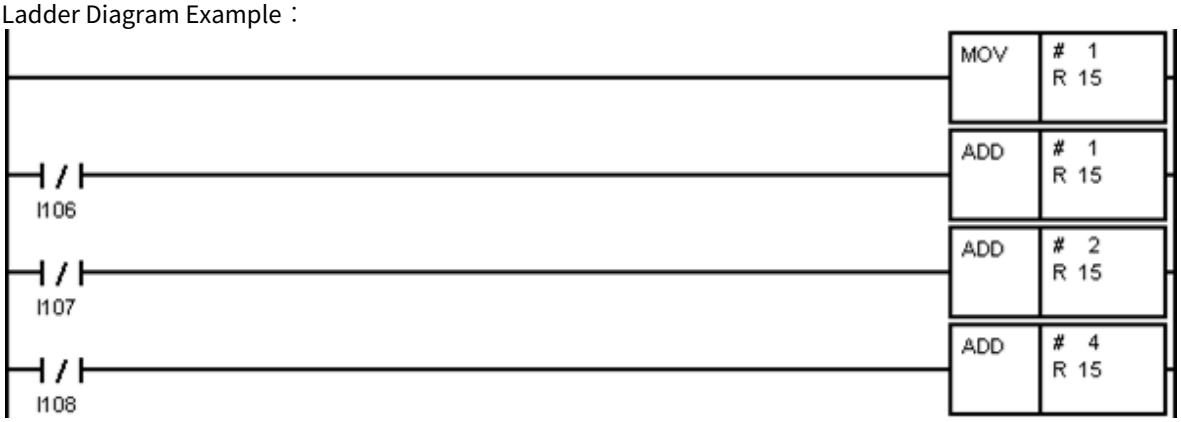
Spindle Override（R15） is composed of three I bits, I-106/ I-107/ I-108, in binary form.

The input/output transfer relationship are shown in the table below.



Input			Output
I-108（+4）	I-107（+2）	I-106（+1）	R15
1	1	1	1+0+0+0=1：50%
1	1	0	1+0+0+1=2：60%
1	0	1	1+0+2+0=3：70%
1	0	0	1+0+2+1=4：80%
0	1	1	1+4+0+0=5：90%
0	1	0	1+4+0+1=6：100%
0	0	1	1+4+2+0=7：110%
0	0	0	1+4+2+1=8：120%

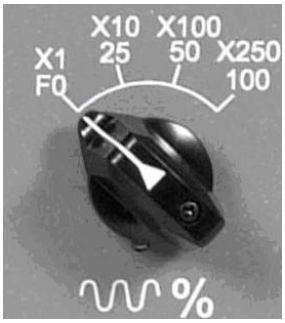
Ladder Diagram Example：



Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

MPG/INJOG Feed（R14） and G00 Override（R18） are composed of two I bits, I-104/ I-105, in binary form.

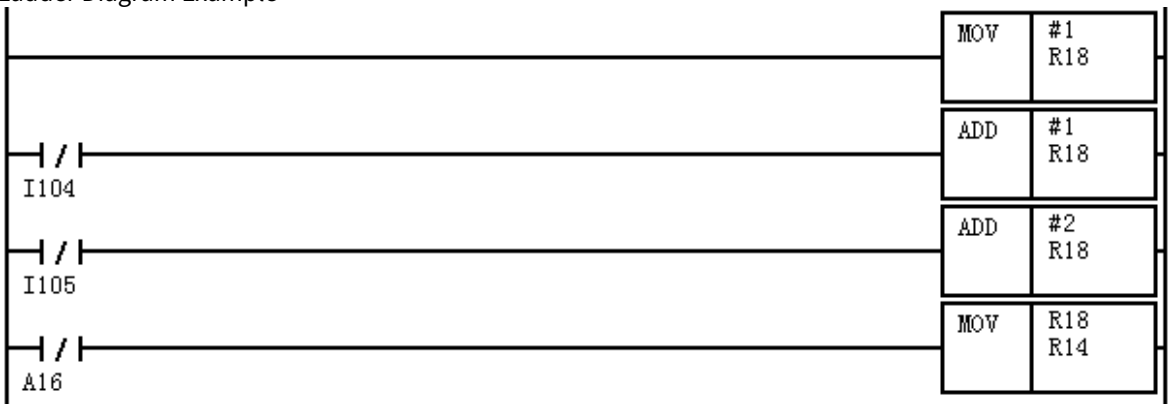
The input/output transfer relationship are shown in the table below.



Input		Output	
I-105（+2）	I-104（+1）	R14	R18
1	1	1+0+0=1：*1	1+0+0=1：F0
1	0	1+0+1=2：*10	1+0+1=2：25%
0	1	1+2+0=3：*100	1+2+0=3：50%
0	0	1+2+1=4：*1000	1+2+1=4：100%

Note：To setup R14, please refer to Pr2001 MPG 4th scaling factor.

Ladder Diagram Example：



Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG

Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

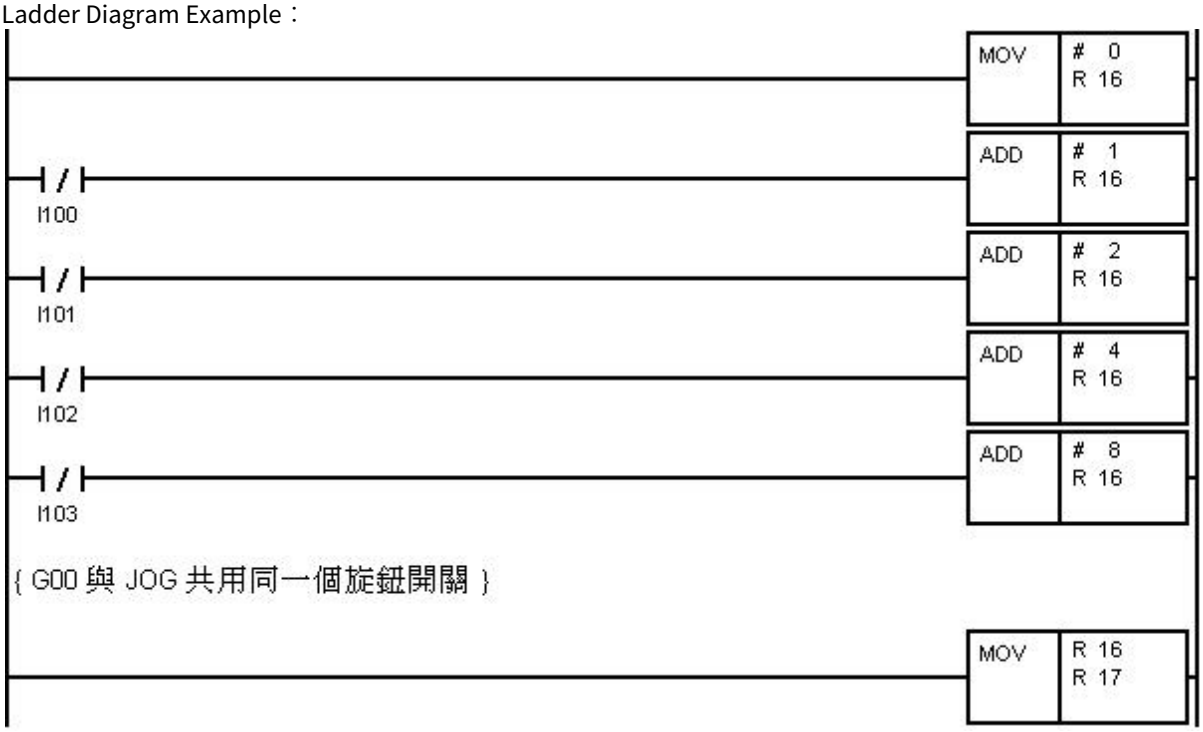
G01（R16） and JOG Override（R17） is composed of 4 I bits, I-100/ I-101/ I-102/ I-103, in binary form.

The input/output transfer relationship are shown in the table below.

Input				Output
I-103（+8）	I-102（+4）	I-101（+2）	I-100（+1）	R16 & R17
1	1	1	1	0+0+0+0+0=0：0%
1	1	1	0	0+0+0+0+1=1：10%
1	1	0	1	0+0+0+2+0=2：20%
1	1	0	0	0+0+0+2+1=3：30%
1	0	1	1	0+0+4+0+0=4：40%
1	0	1	0	0+0+4+0+1=5：50%
1	0	0	1	0+0+4+2+0=6：60%
1	0	0	0	0+0+4+2+1=7：70%
0	1	1	1	0+8+0+0+0=8：80%
0	1	1	0	0+8+0+0+1=9：90%
0	1	0	1	0+8+0+2+0=10：100%
0	1	0	0	0+8+0+2+1=11：110%
0	0	1	1	0+8+4+0+0=12：120%

0	0	1	0	0+8+4+0+1=13：130%
0	0	0	1	0+8+4+2+0=14：140%
0	0	0	0	0+8+4+2+1=15：150%

Ladder Diagram Example：



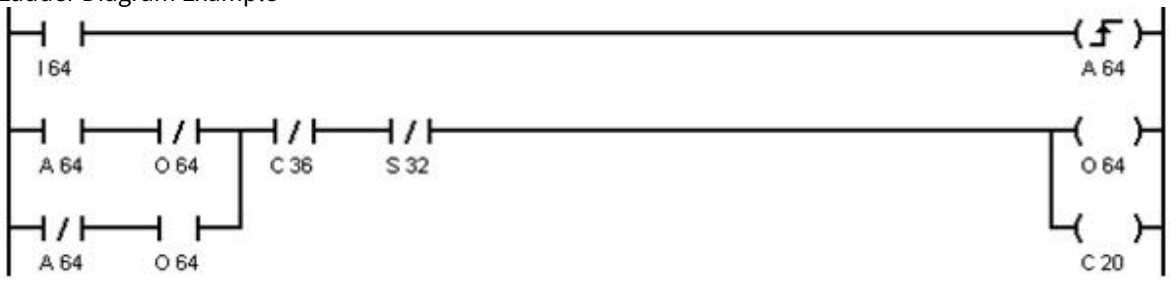
Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

MPG Simulation can only be activated by C20 in the ladder diagram.

When switch to Auto or MDI mode with C20 triggered, G00/G01/G02/G03 speed will be decided by MPG rotation speed after Cycle Start.

The machine moves faster when MPG rotates faster; the machine stops when MPG stops. Applicable in trial processing stage.

Ladder Diagram Example：

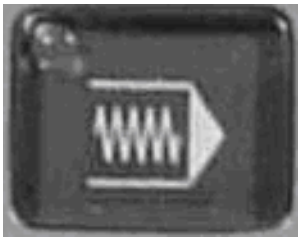


Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

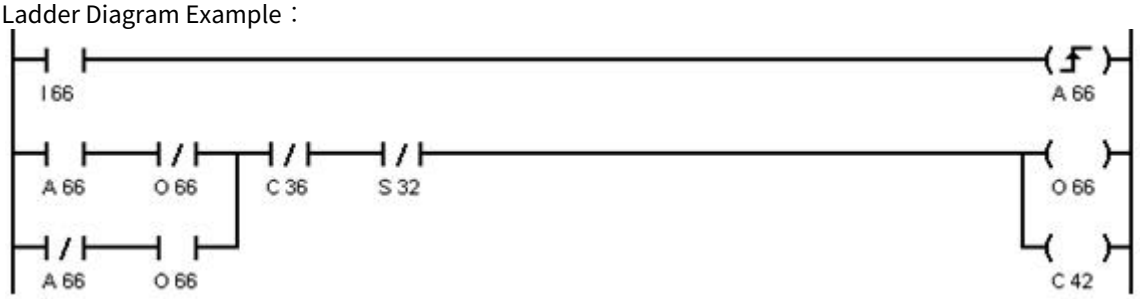
Program Dry Run can only be activated by C42 in the ladder diagram.

When C42 is triggered, the machine will move in dry run speed when executing moving G code.

For G01 block is 5 times the program speed, for G00 block is just G00 speed.



Ladder Diagram Example：

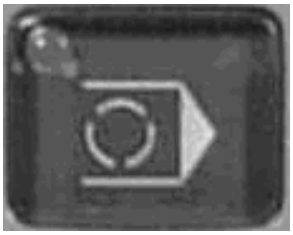


Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

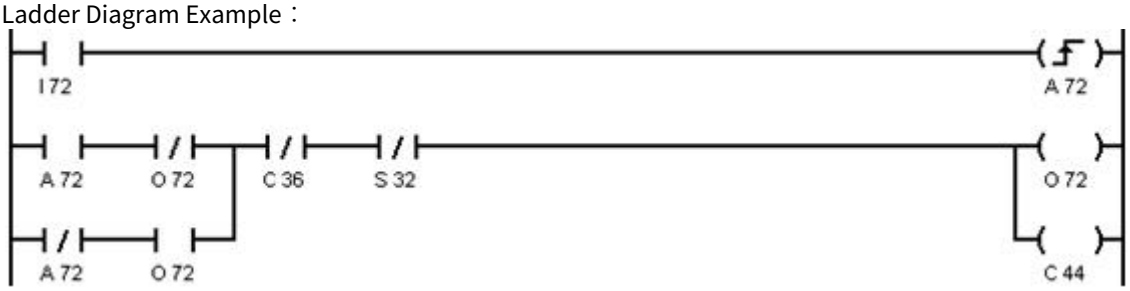
Optional Stop can only be activated by C44 in the ladder diagram.

When C44 is triggered, the controller will switch to B-stop state when executing M01.

When C44 is off, the controller will ignore M01 and proceed to the next block.



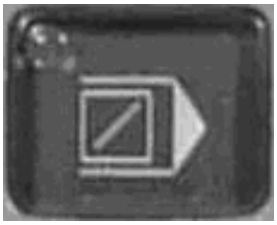
Ladder Diagram Example：



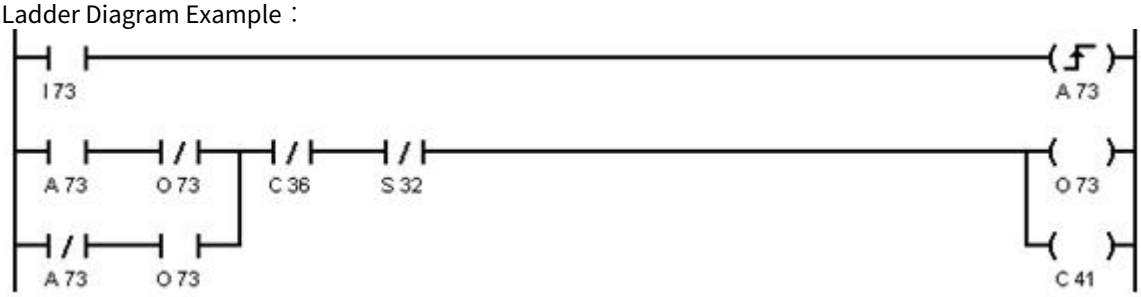
Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

Optional Skip can only be activated by C41 in the ladder diagram.

When C41 is triggered, the controller will skip the block with the skipping symbol "/" and proceed to the next block.



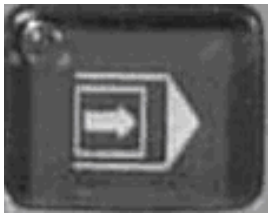
Ladder Diagram Example：



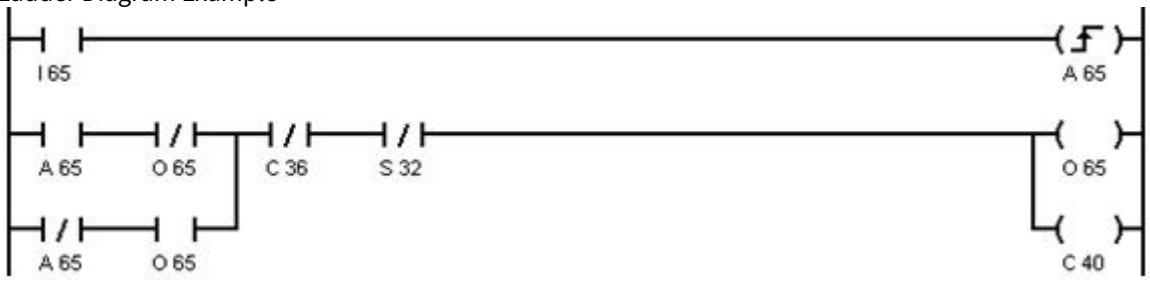
Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

Single Block can only be activated by C40 in the ladder diagram.

When C40 is triggered, the machining process stops every time after finishing a block, and run the next block by pressing Cycle Start.



Ladder Diagram Example：



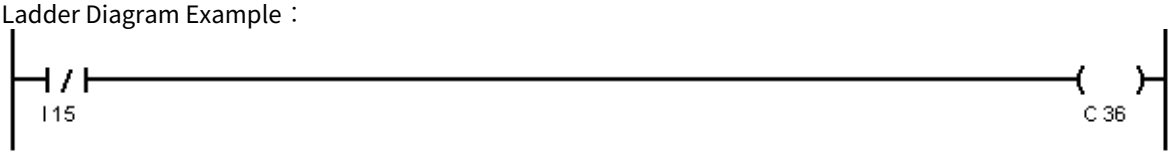
Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

Emergency Stop can only be activated by C36 in the ladder diagram.

When C36 is triggered, the controller will stop all actions and switch to Not Ready state.

After C36 is off, the system restores to Ready state.

Ladder Diagram Example：



Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

SYNTEC

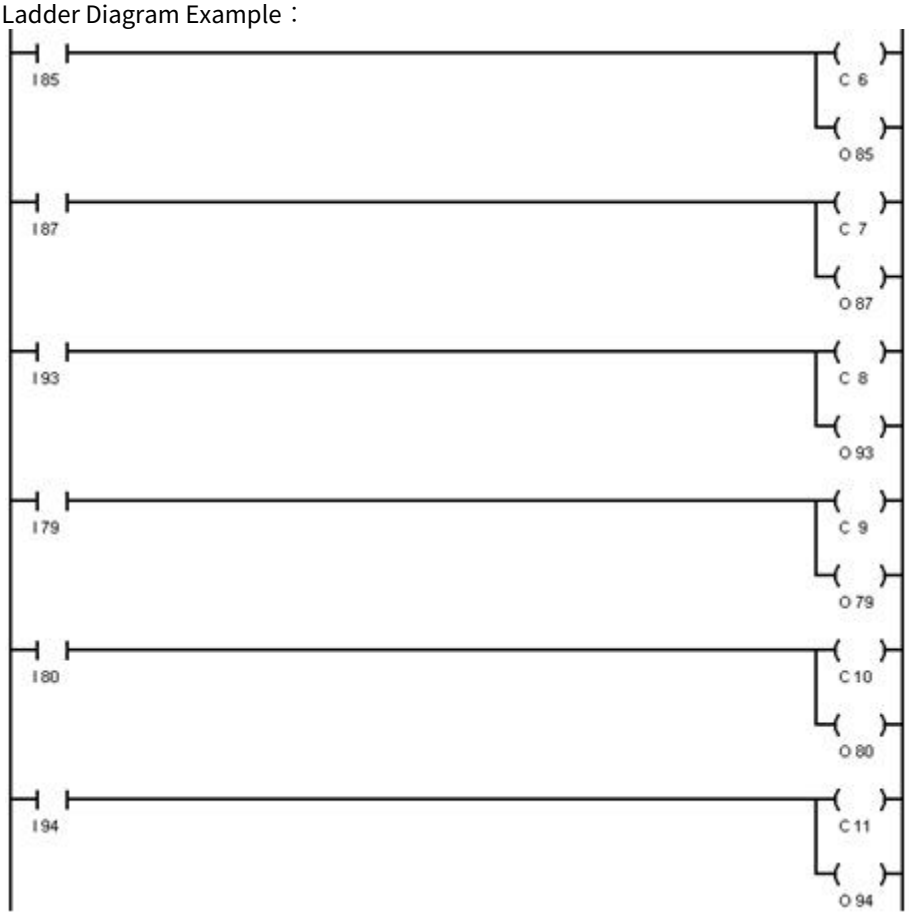
In JOG mode, the controller moves by pressing the arrow keys of corresponding axes and activate the C bit in the table below.

If applied with rapid moving（C23） activated, the axis will be moving in the speed of G00.

PS：The I bit table below is only the corresponding number of one of the standard panel, not all panels are the same.

Axis/Direction	X+	X-	Y+	Y-	Z+	Z-
C Bit	C06	C07	C08	C09	C10	C11
Lathe	I-65	I-71	I-64	I-70	I-69	I-67
Milling Machine	I-85	I-87	I-93	I-79	I-80	I-94

Ladder Diagram Example：



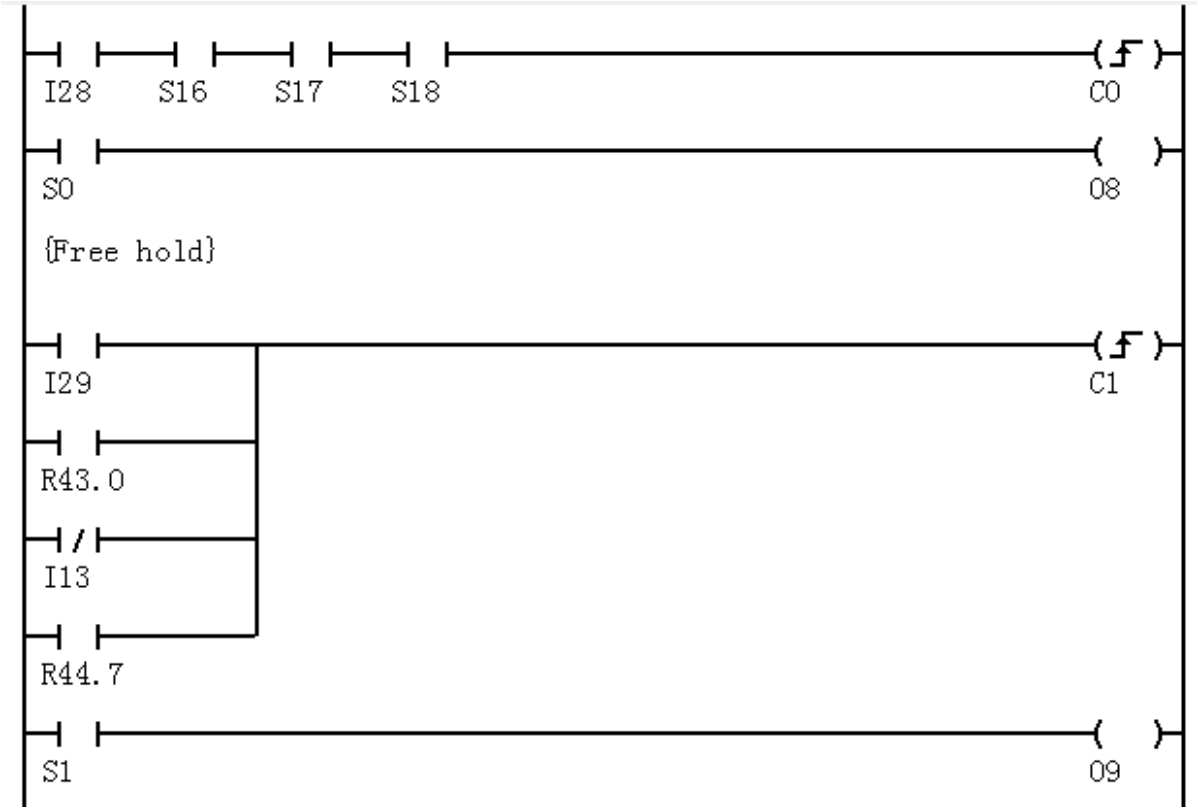
Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

Cycle Start can only be activated by C0 in the ladder diagram; Feed Hold can only be activated by C1 in the ladder diagram.

In Auto or MDI mode, C0 can start the machining process with rising edge-triggered. S0 will be activated after the machining process is started.

C1 can also pause the machining process with rising edge-triggered. S1 will be activated after the process is paused and S0 will be suspended.

Ladder Diagram Example：



Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

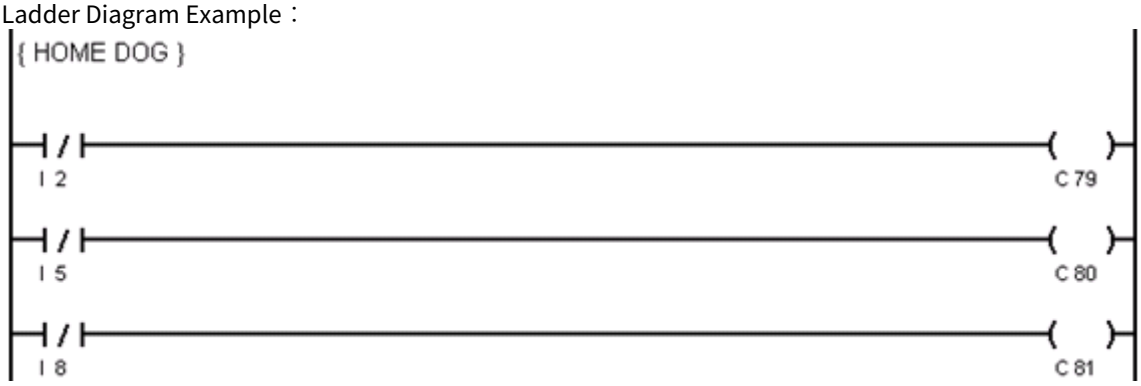
In HOME mode, the controller will move towards the origin with the parameter setups when pressing JOG button of each axis.

When touching a Home Dog（origin switch）, the corresponding C Bit in the table below should be activated.

After the home searching is done, the controller will activate the corresponding home searching complete flag.

	X Axis	Y Axis	Z Axis
Home Dog Signal	C79	C80	C81
Home Searching Complete Flag	S16	S17	S18

Ladder Diagram Example：



Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

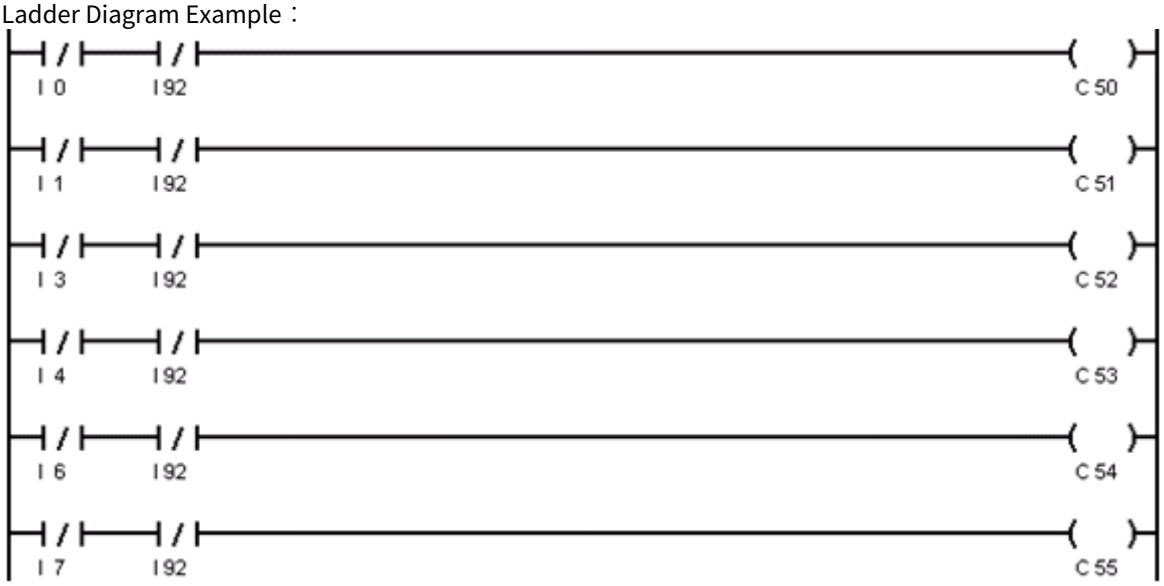
The table below shows the corresponding C Bit of +/- hardware stroke limit for each axis.

When the C Bit is activated, the controller will send alarms and stop the machine automatically.

The controller then can only receive reverse direction commands to bring the machine out of the limit position.

	X+ limit	X- limit	Y+ limit	Y- limit	Z+ limit	Z- lmit
C Bit	C50	C51	C52	C53	C54	C55

Ladder Diagram Example：



PS：Sometimes the hardware limit switch is malfunctioning so the system can't free the alarm.

For situation like this, there should be a way to free the alarm manually, such as I-92 in the picture above.

Mode Selection Spindle Override MPG/ INJOG/ G00 Override G01/ JOG Override MPG Simulation
Program Dry Run Optional Stop Optional Skip Single Block Emergency Stop JOG
Cycle Start/Feed Hold Home Searching Hardware Stroke Limit Spindle CW/CCW/Stop

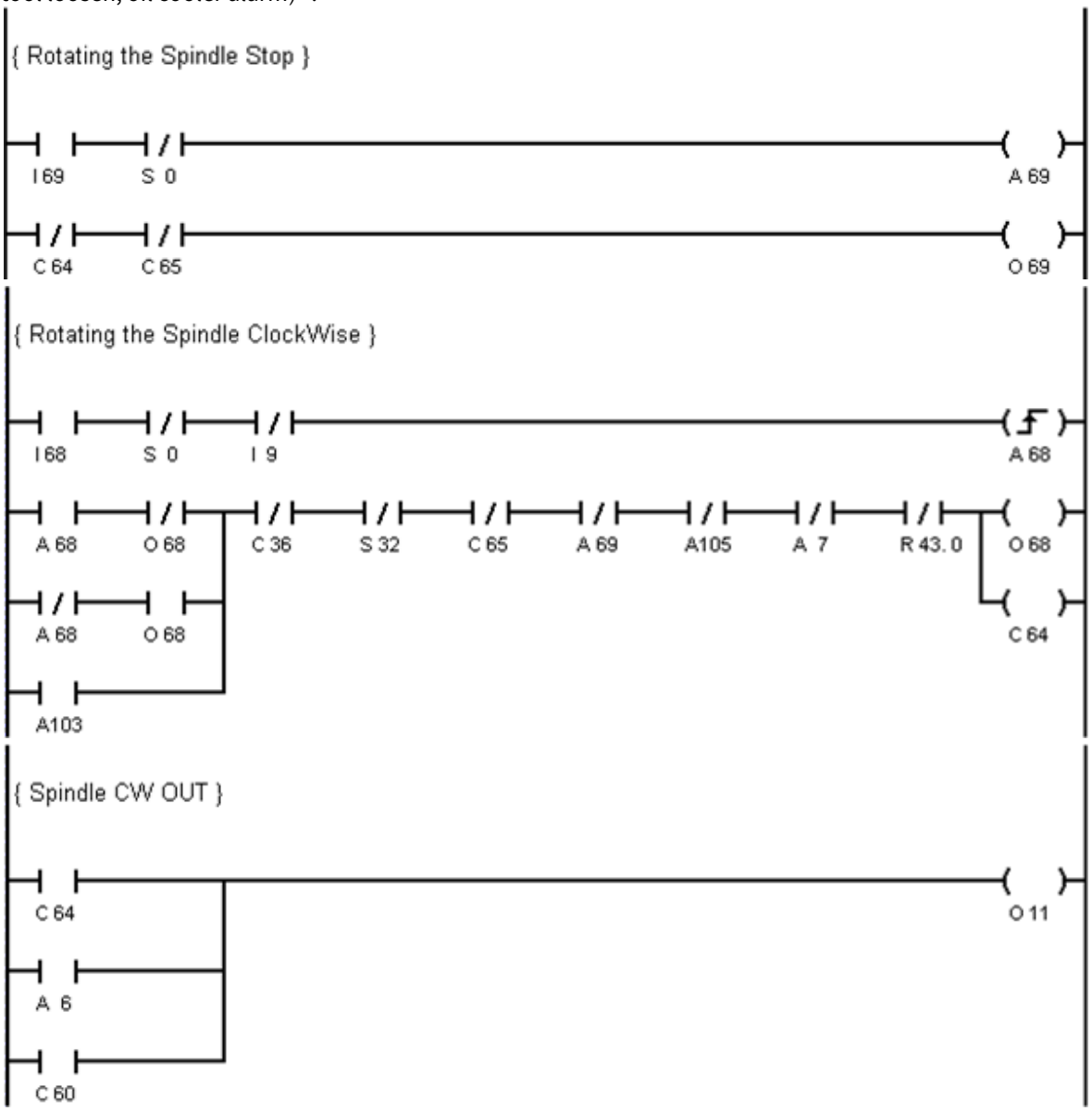
Spindle CW/CCW rotation correspond to C64/C65 respectively, there are some notifications for general operations：

1. C64, C65 must be cut off when pressing the Reset button or the program is running M05, M02 or M30.
2. The spindle shouldn't be rotating if the gripper is not clamping for lathe machine or the tool is loosen for milling machine.
3. Do not stop the spindle manually during the machining（S0 on）, stop the spindle with M05.

Ladder Diagram Example：

I-69 is the spindle stop button on the panel; S32 is the reset signal.

A103 is M03 spindle CW rotation; A105 is M05 spindle stop; A7 is the state spindle rotation is banned（like tool loosen, oil cooler alarm）.



1.4 Application Example of Carousel Tool Magazine in Milling Machine

M Code T Code S Code

1.4.1 M Code

M Code Definition Table

M code below should be provided in ladder diagram of standard milling machine：

M code	Function
M03	Spindle CW Rotation
M04	Spindle CCW Rotation
M05	Spindle Stop
M07	solenoid valve of air blow ON
M08	cutting fluid motor ON
M09	solenoid valve of air blow & cutting fluid motor OFF
M19	Spindle Orientation
M30	solenoid valve of air blow, cutting fluid motor OFF, Spindle Stop
M37	Chip Conveyor CW Rotation
M38	Chip Conveyor CCW Rotation
M50	Tool Unclamp
M51	Tool Clamp
M52	Tool Magazine Out
M53	Tool Magazine In

PS：M01（Optional Stop）, M02（Program Finish）, M98（Call Subprogram）, M99（Program Return）.

These four M code are provided by the controller kernel, so they're not listed above.

M Code Operation Specification

1. M03（Spindle CW Rotation）：This M code is used to make the spindle do CW rotation, apply with S code to set rotation speed. Specification as follows：
    a. When executing M03, proceed the NC program when the spindle speed（R36） reaches 80% target speed（R871~）.
    b. During spindle CW rotation signal（C64 On） output, keep checking if the spindle speed is higher than 70% target speed, if not, the machining process will be paused immediately and the alarm will be sent.
2. M04（Spindle CCW Rotation）：This M code is used to make the spindle do CWW rotation, apply with S code to set rotation speed. Specification as follows：
    a. When executing M04, proceed the NC program when the spindle speed（R36） reaches 80% target speed（R871~）.
    b. During spindle CCW rotation signal（C65 On） output, keep checking if the spindle speed is higher than 70% target speed, if not, the machining process will be paused immediately and the alarm will be sent.
3. M05（Spindle Stop）：This M code is used to stop the spindle rotation, also able to cancel the spindle orientation. Specification as follows：
    a. When executing M05, turn off C64, C65 to stop the spindle.
    b. When the spindle speed（R36） is lower than 10 RPM, the spindle control will be turned off.（applicable for pulse spindle）
4. M19（Spindle Orientation）：This M code is used to make the spindle process the orientation. Specification as follows：
    a. When executing M19, turn on the spindle orientation signal（C61）.
    b. Proceed the NC program after the controller receive spindle orientation complete signal（S61）.
5. M50（Tool Unclamp）：This M code is used to unclamp the tool. Specification as follows：
    a. When executing M50, output tool unclamp signal.
    b. Proceed the NC program after the controller receive tool unclamp complete signal.
6. M51（Tool Clamp）：This M code is used to clamp the tool. Specification as follows：
    a. When executing M51, turn off the tool unclamp signal.
    b. Proceed the NC program after the controller receive tool clamp complete signal.
7. M52（Tool Magazine Out）：This M code is used to push the tool magazine out. Specification as follows：
    a. When executing M52, output push out signal.
    b. Proceed the NC program after the controller receive tool magazine out signal.
8. M53（Tool Magazine In）：This M code is used to pull the tool magazine in. Specification as follows：
    a. When executing M53, turn off the push out signal.
    b. Proceed the NC program after the controller receive tool magazine in signal.

M Code Technical Specification

1. M03（Spindle CW Rotation）：
    a. M03 can only be executed when the tool is clamped.

   b. If the spindle speed（R36） couldn't reach 80% target speed（R871~） within 10 seconds, "Spindle CW Rotation Speed Error" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
   c. Keep checking the spindle speed when the spindle is rotating, if the spindle speed is lower than 70% target speed, "Spindle Speed too Low" Alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
2. M04（Spindle CCW Rotation）：
   a. M04 can only be executed when the tool is clamped.
   b. If the spindle speed（R36） couldn't reach 80% target speed（R871~） within 10 seconds, "Spindle CCW Rotation Speed Error" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
   c. Keep checking the spindle speed when the spindle is rotating, if the spindle speed is lower than 70% target speed, "Spindle Speed too Low" Alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
3. M05（Spindle Stop）：
   a. No need to wait when executing M05, which means C38 should stay triggered.
   b. Turn off the external spindle execution（RUN） signal when the spindle speed is lower than 10 RPM.（Applicable for pulse spindle）
4. M19（Spindle Orientation）：
   a. M19 can only be executed when the tool is clamped.
   b. If the spindle orientation complete signal is not sent back to the controller within 3 seconds, "Spindle Orientation Failure" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
   c. During the spindle orientation, if the orientation complete signal sent from the inverter is interrupted abnormally, "Spindle Orientation Abnormal Interruption" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
   d. During the spindle orientation, it could be cancelled by M03, M04, M05 and start the next action directly.
5. M50（Tool Unclamp）：
   a. Do not unclamp the tool when spindle is rotating.
   b. Only unclamp the tool when the spindle speed（R36） is lower than 10 RPM.
   c. If the tool unclamp signal is not sent back to the controller within 3 seconds, "Tool Unclamp Failure" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
   d. The tool unclamp action is completed only if the tool unclamp signal lasts over 0.1 second.
6. M51（Tool Clamp）：
   a. Do not clamp the tool when spindle is rotating.
   b. Only clamp the tool when the spindle speed（R36） is lower than 10 RPM.
   c. If the tool clamp signal is not sent back to the controller within 3 seconds, "Tool Clamp Failure" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
   d. The tool clamp action is completed only if the tool clamp signal lasts over 0.1 second.
7. M52（Tool Magazine Out）：
   a. Do not push the tool magazine out when spindle is rotating.
   b. Only push the tool magazine out when the spindle speed（R36） is lower than 10 RPM.
   c. If the "tool magazine out" signal is not sent back to the controller within 10 seconds, "Tool Magazine Push Out Failure" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.

d. The pushing out action is completed only if the "tool magazine out" signal lasts over 0.1 second.

8. M53（Tool Magazine In）：
    a. Do not pull the tool magazine in when spindle is rotating.
    b. Only pull the tool magazine in when the spindle speed（R36） is lower than 10 RPM.
    c. If the "tool magazine in" signal is not sent back to the controller within 10 seconds, "Tool Magazine Pull In Failure" alarm will be sent and the machining process will be paused. Press "Reset" button to cancel the alarm.
    d. The pulling in action is completed only if the "tool magazine in" signal lasts over 0.1 second.
9. Others：
    a. Can't control spindle manually by operation panel during the machining process.
    b. If a G code and M05 are in the same block, M05 can only be executed after G code is completed（S30 activated）.
10. Notification：
    a. After dealing with the tool changing failure alarm, please make sure if the magazine tool number and spindle tool number recorded by the controller is the same as the actual tool number. If not, please correct the tool number table then reboot the controller.

M Code T Code S Code

1.4.2 T Code

T Code Operation Specification

When executing T code in NC program, the tool will be changed according to the process below：

1. Spindle stop（M05）
2. Tool magazine rotates to current tool number
3. Spindle Orientation（M19）
4. Z axis moves to 2nd reference point by G00（Pr2803）
5. Tool magazine push out（M52）
6. Tool unclamp（M50）
7. Z axis moves to mechanical origin by G00
8. Tool magazine rotates to target tool number
9. Z axis moves to 2nd reference point by G00（Pr2803）
10. Tool Clamp（M51）
11. Tool Magazine pull in（M53）
12. Cancel Spindle Orientation（M05）
13. Modify Spindle Tool Number
14. Continue to execute NC Program

T Code Technical Specification

1. When the tool magazine is out, the spindle and the tool magazine are not allowed to rotate.
2. If the tool count signal is not sent to the controller within 3 seconds after activate tool magazine rotation, "Tool Count Error" alarm will be sent and the machining process will be paused.
3. During tool changing process, if "Spindle Orientation Failure", "Tool Magazine Push Out Failure", "Tool Unclamp Failure", "Tool Clamp Failure", "Tool Magazine Pull In Failure" or "Tool Count Error" alarm happens, "Tool Change Failure" alarm will be sent and the machining process will be paused. Please wait till the tool clamp signal and "tool magazine in" signal appear then press "Reset" button to cancel the tool changing process.

T Code Macro Calling Function

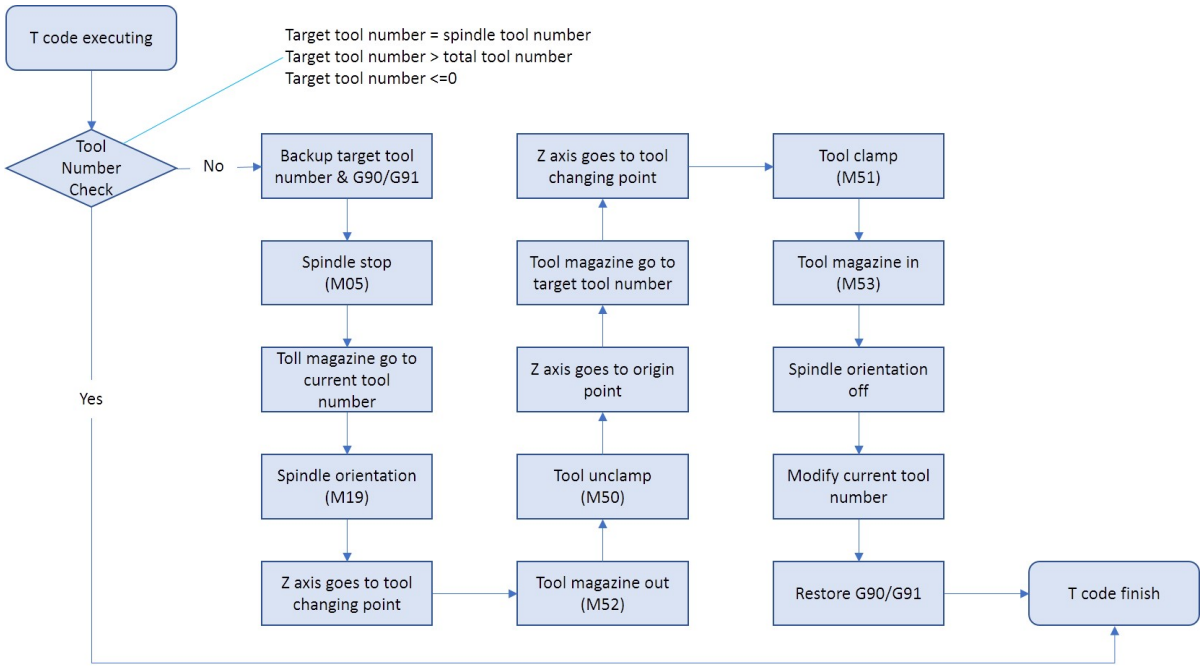
Tool changing is composed of several actions, it would be better to execute it as a T code macro.

After finishing the setups below, it will enter T0000 macro first when executing T code, then triggers the related PLC actions according to T0000.

1. Set Pr3215 to 1.

2. Get the T code macro ready and save it in the controller. The file name of the macro should be T0000（no extension）.

MSD of T code



Example of T0000 Macro

```
%@MACRO
// @481=R81_total  tool number
//@502=R102_current spindle tool number
// #1004_G90/G91 mode；#1036_target tool number；#26023=Pr2803_tool changing point of Z axis
IF ( @502 = #1036 OR #1036 = 0 OR @481 < #1036 ) THEN
 M99;  //If target tool number = current tool number or tool No.0 or larger than total tool number, leave T0000
END_IF;
@102:=#1036;  // save target tool number
@103:=#1004;  // save G90/G91 state
M3S0;  // stop the spindle by S0, then use M05
M05;
WAIT();
T@502;  // tool magazine rotates to current tool number
```

```
M19;  // spindle orientation
G90 G53 Z#26023;  // Z axis moves to tool changing point（Pr2803）
G4X0.5;
WAIT();
M52;  // tool magazine push out
M50;  // tool unclamp
G90 G53 Z0;  // Z axis moves to mechanical origin
WAIT();
T@102;  // tool magazine rotates to target tool number
G90 G53 Z#26023;  // Z xis moves to tool changing point（Pr2803）
G4X0.5;
WAIT();
M51;  // tool clamp
M53;  // tool magazine pull in
M05;  // cancel spindle orientation
WAIT();
@502:=@102;  // modify current tool number
G@103;  // restore G90/G91 state
M99;
```

M Code T Code S Code

1.4.3 S Code

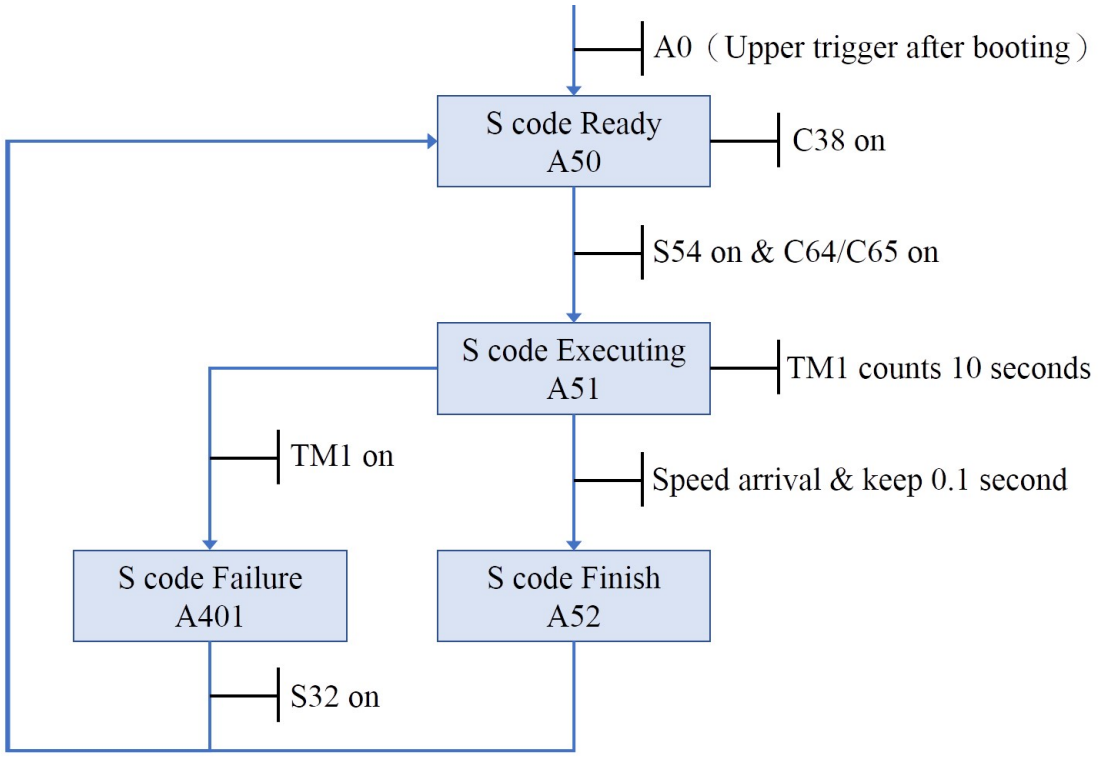
S Code Operation Specification

1. Executing S code while the spindle is rotating will change the spindle speed.
2. Proceed the NC program if the spindle speed（R36） is higher than 80% target speed（R871~）and lower than 110% target speed.

S Code Technical Specification

1. If the spindle speed（R36） couldn't reach 80% ~ 110% target speed within 10 seconds, "Spindle Speed Change Failure" alarm will be sent and the machining process will be paused.
2. The spindle speed changing is completed only if the spindle speed stays between 80% ~ 110% target speed over than 0.1 second.

MSD of S Code



1.5 Common C/S/R

1.5.1 Axial Related

Function	Item	Type	Explanation	1	2	3	4	5	6	7	8
				9	10	11	12	13	14	15	16
Axis Flag & Register	1	C	Axis JOG（+）	C6	C8	C10	C170	C172	C12	C174	C176
				C178	C180	C182	C184	C186	C188	C190	C192
	2	C	Axis JOG（-）	C7	C9	C11	C171	C173	C13	C175	C177
				C179	C181	C183	C185	C187	C189	C191	C193
	3	C	MPG axis selection	C16	C17	C18	C215	C216	C19	C217	C218
				C219	C220	C221	C222	C223	C224	C225	C226
	4	C	Setup machine coordinate	C25	C26	C27	C230	C231	C28	C232	C233
				C234	C235	C236	C237	C238	C239	C240	C241
	5	C	Manual control (Monitor Mode)	C31	C32	C33	C245	C246	C34	C247	C248
				C249	C250	C251	C252	C253	C254	C255	C256
	6	C	Positive hardware stroke Limit	C50	C52	C54	C140	C142	C56	C144	C146

				C148	C150	C152	C154	C156	C158	C160	C162
	7	C	Negative hardware stroke limit	C51	C53	C55	C141	C143	C57	C145	C147
				C149	C151	C153	C155	C157	C159	C161	C163
	8	C	PLC axis control flag	C66	C67	C68	C260	C261	C69	C262	C263
				C264	C265	C266	C267	C268	C269	C270	C271
	9	C	Home dog signal Flag	C79	C80	C81	C200	C201	C82	C202	C203
				C204	C205	C206	C207	C208	C209	C210	C211
	10	S	Axis busy flag	S6	S7	S8	S155	S156	S9	S157	S158
				S159	S160	S161	S162	S163	S164	S165	S166
Function	Item	Type	Explanation	1	2	3	4	5	6	7	8
				9	10	11	12	13	14	15	16
Axis Flag & Register	11	S	Axis home searching finish flag (Home OK)	S16	S17	S18	S140	S141	S19	S142	S143
				S144	S145	S146	S147	S148	S149	S150	S151
	12	S	Servo alarm flag	S171	S172	S173	S174	S175	S176	S177	S178
				S179	S180	S181	S182	S183	S184	S185	S186

13	R	Axis program coordinate	R721 (R26)	R722 (R27)	R723 (R28)	R724	R725	R726 (R29)	R727	R728
			R729	R730	R731	R732	R733	R734	R735	R736
14	R	Axis machine coordinate	R741 (R31)	R742 (R32)	R743 (R33)	R744	R745	R746 (R34)	R747	R748
			R749	R750	R751	R752	R753	R754	R755	R756
15	C	Machine lock for single axis	R600.1	R600.2	R600.3	R600.4	R600.5	R600.6	R600.7	R600.8
			R600.9	R600.10	R600.11	R600.12	R600.13	R600.14	R600.15	R600.16
16	C	MPG simulation for single axis (PLC Axis)	R601.1	R601.2	R601.3	R601.4	R601.5	R601.6	R601.7	R601.8
			R601.9	R601.10	R601.11	R601.12	R601.13	R601.14	R601.15	R601.16
17	S	G31 skip position latched	R610.1	R610.2	R610.3	R610.4	R610.5	R610.6	R610.7	R610.8
			R610.9	R610.10	R610.11	R610.12	R610.13	R610.14	R610.15	R610.15
18	S	Axis moving flag	R611.1	R611.2	R611.3	R611.4	R611.5	R611.6	R611.7	R611.8
			R611.9	R611.10	R611.11	R611.12	R611.13	R611.14	R611.15	R611.16
19	C	Positive machine lock	R612.1	R612.2	R612.3	R612.4	R612.5	R612.6	R612.7	R612.8

				R612.9	R612.10	R612.11	R612.12	R612.13	R612.14	R612.15	R612.16
	20	C	Negative machine lock	R613.1	R613.2	R613.3	R613.4	R613.5	R613.6	R613.7	R613.8
				R613.9	R613.10	R613.11	R613.12	R613.13	R613.14	R613.15	R613.16
	21	S	M code read flag for PLC axis	R629.1	R629.2	R629.3	R629.4	X	X	X	X
				X	X	X	X	X	X	X	X
Function	Item	Type	Explanation	1	2	3	4	5	6	7	8
				9	10	11	12	13	14	15	16
Axis Flag & Register	22	C	M code finish flag for PLC axis	R630.1	R630.2	R630.3	R630.4	X	X	X	X
				X	X	X	X	X	X	X	X
	23	R	M code content for PLC axis	R681	R682	R683	R684	R685	R686	R687	R688
				R689	R690	R691	R692	R693	R694	R695	R696
	24	R	G00/G01 override for PLC axis	R661	R662	R663	R664	R665	R666	R667	R668
				R669	R670	R671	R672	R673	R674	R675	R676
	25	R	MPG command multiplier ratio of each axis	R641	R642	R643	R644	R645	R646	R647	R648
				R649	R650	R651	R652	R653	R654	R655	R656
	26	R	Axis speed of each axis	R701	R702	R703	R704	R705	R706	R707	R708
				R709	R710	R711	R712	R713	R714	R715	R716

27	R	Machine coordinate of break point	R861	R862	R863	R864	R865	R866	R867	R868
			R869	R870	X	X	X	X	X	X
28	R	Temperature compensation of each axis	R901	R902	R903	R904	R905	R906	R907	R908
			R909	R910	R911	R912	R913	R914	R915	R916
29	R	Home dog shift	R961	R962	R963	R964	R965	R966	R967	R968
			R969	R970	R971	R972	R973	R974	R975	R976
30	R	Absolute counter of each axis	R981	R982	R983	R984	R985	R986	R987	R988
			R989	R990	R991	R992	R993	R994	R995	R996
31	R	Following error of each axis	R5001	R5002	R5003	R5004	R5005	R5006	R5007	R5008
			R5009	R5010	R5011	R5012	R5013	R5014	R5015	R5016

1.5.2 MST Channel and Multi Axis Group

Function	Item	Type	Explanation	Channel 1	Channel 2	Channel 3	Channel 4
MST Channel & Multi Axis Group	1	C	M.S.T code finish	C38	R615.0	R619.0	R623.0
	2	S	Finish signal of G code in same block (Distribution End)	S30	R615.1	R619.1	R623.1
	3	S	M code read (Multiple M code 1)	S29	R615.2 (R615.24)	R619.2 (R619.24)	R623.2 (R623.24)
	4	S	M code read (Multiple M code 2)	S11	R615.25	R619.25	R623.25
	5	S	M code read (Multiple M code 3)	S12	R615.26	R619.26	R623.26
	6	S	M code read (Multiple M code 4)	S13	R615.27	R619.27	R623.27
	7	S	M code read (Multiple M code 5)	S14	R615.28	R619.28	R623.28
	8	S	S code Read	S54	R615.3	R619.3	R623.3
	9	S	T code Read	S69	R615.4	R619.4	R623.4
	10	S	MST flag sent from 1st axis group	R589.1	R615.8	R619.8	R623.8
	11	S	MST flag sent from 2nd axis group	R589.2	R615.9	R619.9	R623.9
	12	S	MST flag sent from 3rd axis group	R589.3	R615.10	R619.10	R623.10

	13	S	MST flag sent from 4th axis group	R589.4	R615.11	R619.11	R623.11
	14	S	MST related information of multi axis group for 1st spindle	R589.11	R615.16	R619.16	R623.16
	15	S	MST related information of multi axis group for 2nd spindle	R589.12	R615.17	R619.17	R623.17
	16	S	MST related information of multi axis group for 3rd spindle	R589.13	R615.18	R619.18	R623.18
	17	S	MST related information of multi axis group for 4th spindle	R589.14	R615.19	R619.19	R623.19
	18	R	M code content (Multiple M code 1)	R1（R2050）	R616（R2060）	R620（R2070）	R624（R2080）
	19	R	M code content (Multiple M code 2)	R2051	R2061	R2071	R2081
	20	R	M code content (Multiple M code 3)	R2052	R2062	R2072	R2082
	21	R	M code content (Multiple M code 4)	R2053	R2063	R2073	R2083
	22	R	M code content (Multiple M code 5)	R2054	R2064	R2074	R2084
Function	Item	Type	Explanation	Channel 1	Channel 2	Channel 3	Channel 4

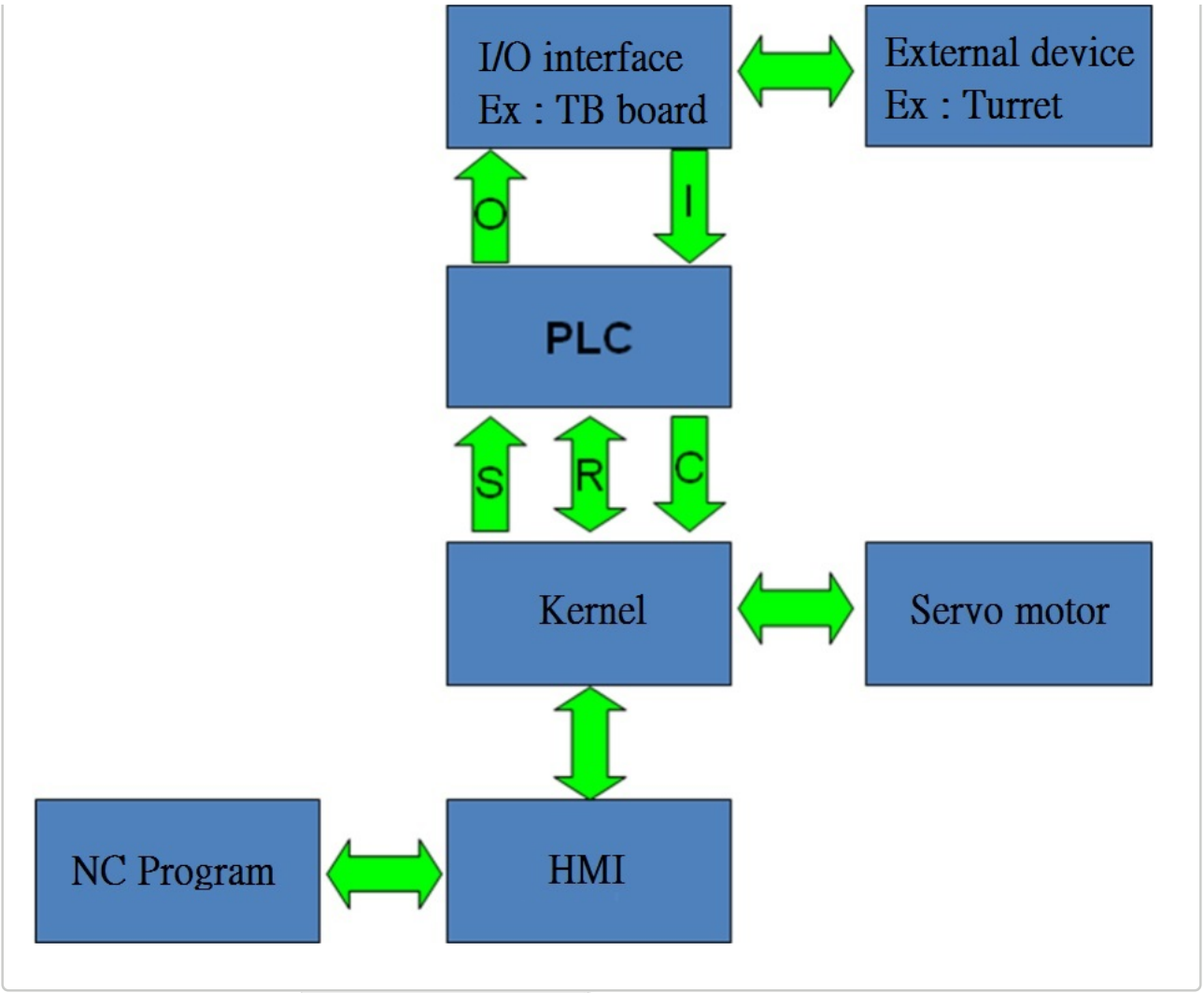
MST Channel & Multi Axis Group	23	R	S Code Content	R2	R617	R621	R625
	24	R	T Code Content	R3	R618	R622	R626
	25	R	Spindle selection for multi axis group	R791	R792	R793	R794
	26	R	Cause of block stop	R841	R842	R843	R844
	27	R	Maximum spindle cutting speed	R5021	R5022	R5023	R5024
	28	R	Active session ID	R21	R21	R21	R21
	29	R	B Code Content	R5	Not Supporting	Not Supporting	Not Supporting
	30	S	B Code Read	S53	Not Supporting	Not Supporting	Not Supporting

2 PLC Components Introduction.

2.1 Introduction of Components in Ladder Diagram

Syntec controller mainly provides I, O, A, C, S, R, CT, TM 8 kinds of components in ladder diagram for program editors, the details are listed in table below：

Name	Definition	Specification	Total Amount	Note
I	Entity input point	The specifications and number are related to the chosen I/O card.		
O	Entity output point	Only On/Off.		
A	Virtual auxiliary contact point	Only On/Off	A0~A511, total 512 points	
C	Command sent from ladder diagram to kernel	Only On/Off	C0~C511, total 512 points	
S	System state sent from kernel to ladder diagram	Only On/Off	S0~S511, total 512 points	
R	Registers for communication between ladder diagram and kernel	32 bits register, can store values or used as binary bits	R0~R65535, total 65536 points	It only supports R0~R7999 for DOS system.
CT	Virtual auxiliary counter	upward /downward/ring counter	CT0~CT255, total 256 points	
TM	Virtual auxiliary timer	4 units：1 second/0.1 second/ 0.01 second/0.001 second	TM0~TM255, total 256 points	


I/O interface
Ex : TB board
External device
Ex : Turret
O
I
PLC
S
R
C
Kernel
Servo motor
NC Program
HMI

2.2 Instruction of PLC Editor Components

2.2.1 Contact

Normal Open Contact

1. Figure:

		
General contact	Timer contact	Counter contact

2. Description: So called "A contact". Normal Open Contact is usually open, and become close when triggered.
3. Amount: 512 points for each I/ O/ C/ S/ A; 256 points for both timer and counter; and only R0~R639 are allowed to use contact component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

Normal Close Contact

1. Figure:

		
General contact	Timer contact	Counter contact

2. Description: So called "B contact". Normal Close Contact is usually close, and become open when triggered.
3. Amount: 512 points for each I/ O/ C/ S/ A; 256 points for both timer and counter; and only R0~R639 are allowed to use contact component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

Connector

1. Figure：

	
Horizontal connector	Vertical connector

2. Description：Used to connect with other components or fill in the blank part of ladder diagram.

2.2.2 Coil

Normal Open Coil

1. Figure

Normal open coil

2. Description: When input signal is ON, output ON; when input signal is OFF, output OFF.
3. Amount: 512 points for each O/ C/ A; and only R0~R639 are allowed to use coil component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

Normal Close Coil

1. Figure

Normal close coil

2. Description: When input signal is ON, output OFF; when input signal is OFF, output ON.
3. Amount: 512 points for each O/ C/ A; and only R0~R639 are allowed to use coil component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

Positive Edge Triggered Coil

1. Figure

Positive edge triggered coil

2. Description: Outputs a pulse signal when the input signal changes from OFF to ON. The cycle time of the pulse signal equals to the scan time of ladder diagram (Pr3204).
3. Amount: 512 points for each O/ C/ A; and only R0~R639 are allowed to use coil component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

Negative Edge Triggered Coil

1. Figure


Negative edge triggered coil

2. Description: Outputs a pulse signal when the input signal changes from ON to OFF. The cycle time of the pulse signal equals to the scan time of ladder diagram (Pr3204).
3. Amount: 512 points for each O/ C/ A; and only R0~R639 are allowed to use coil component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

Latch Coil

1. Figure


Latch Coil

2. Description: When the input signal is ON, the output is ON ans stays ON.
3. Amount: 512 points for each O/ C/ A; and only R0~R639 are allowed to use coil component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

Unlatch Coil

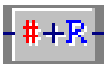
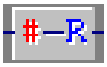
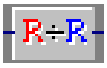
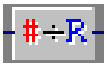
1. Coil


Unlatch coil

2. Description: When the input signal is ON, the output is OFF and stays OFF.
3. Amount: 512 points for each O/ C/ A; and only R0~R639 are allowed to use coil component because the registers in this range can do 32 bit access in ladder diagram, for example: R50.0 ~ R50.31.

2.2.3 Arithmetic

1. Figure

	
R2 = R2 + R1	R2 = R2 + #
	
R2 = R2 – R1	R2 = R2 - #
	
R2 = R2 × R1	R2 = R2 × #
	
R2 = R2 ÷ R1	R2 = R2 ÷ #

PS: Left side of the figure: R1; right side of the figure: R2

2. Description:
   a. Run the arithmetic operations when the input signal is ON.
   b. # represents constant, range: -32768 ~ 32767.
   c. The component supports R bit register indirect addressing function.

2.2.4 Compare

1. Figure

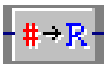
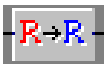
	
Is R1 bigger than R2	Is # bigger than R2
	
Is R1 smaller than R2	Is # smaller than R2
	
Is R1 equal to R2	Is # equal to R2

PS: Left side of the figure: R1; right side of the figure: R2

2. Description:
   a. When the comparing result is true, output signal is ON; when the comparing result is false, output signal is OFF.
   b. # represents constant, range: -32768 ~ 32767.
   c. The component supports R bit register indirect addressing function.

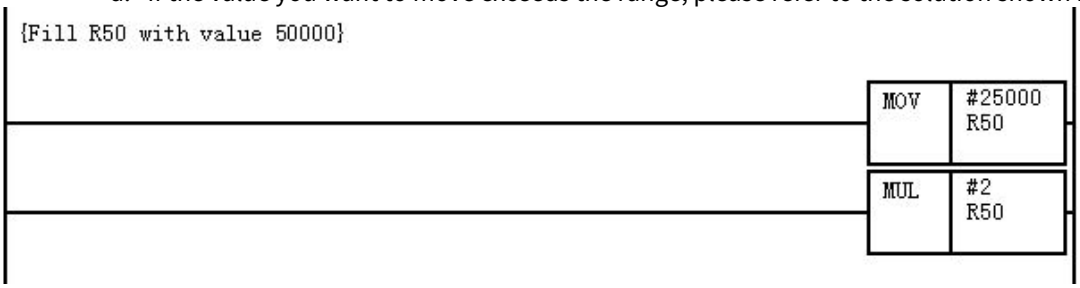
2.2.5 Move

1. Figure

	
Move # to R2	Move R1 to R2

PS：Left side of the figure: R1; right side of the figure: R2

2. Description:
   a. When the input signal is ON, start the moving action.
   b. # represents constant, range: -32768 ~ 32767.
   c. The component supports R bit register indirect addressing function.
   d. If the value you want to move exceeds the range, please refer to the solution shown below:



2.2.6 Logic

1. Figure

	
R2 = R2 AND R1	R2 = R2 AND #
	
R2 = R2 OR R1	R2 = R2 OR #
	
R2 = R2 XOR R1	R2 = R2 XOR #

PS: Left side of the figure :R1; right side of the figure: R2

2. Description: When the input signal is ON, run the logic operation. Every R bit register is in 32 bit form, so each bit runs logic operation separately.

AND Truth Table		
R1	R2	Result
Off	Off	Off
Off	On	Off
On	Off	Off
On	On	On

OR Truth Table		
R1	R2	Result
Off	Off	Off
Off	On	On
On	Off	On
On	On	On

XOR Truth Table		
R1	R2	Result
Off	Off	Off
Off	On	On
On	Off	On
On	On	Off

2.2.7 Timer

1. Figure

constant type	register type
	
Unit: 0.001 second	Unit: 0.001 second
	
Unit: 0.01 second	Unit: 0.01 second
	
Unit: 0.1 second	Unit: 0.1 second
	
Unit: 1 second	Unit: 1 second

2. Description:
   a. When the input signal changes from OFF to ON, the timer returns to 0. When the input signal stays ON, the timer starts counting.
   b. When the timer counts to the target time, the output signal is ON and the corresponding timer contact is also ON. When the input signal is OFF, the timer stops counting, the output signal is OFF and the corresponding timer contact is OFF.
   c. When using the register type timers, if the register value changes, the target time will be updated only when the input signal turns from OFF to ON and restart the counting.
3. Amount: Total 256 timers for use（TM0~TM255）

2.2.8 Counter

One Way Counter

1. Figure

	
Upward counter of constant type	Upward counter of register type
	
Downward counter of constant type	Downward counter of register type

2. Description:
   a. When the input signal changes from OFF to ON, the counter will add (or subtract) 1 time till reaching the target value.
   b. When the counter reaches the target value, the output signal is ON and its corresponding counter contact is also ON.
   c. The initial value of upward counters is 0, the target value is the value of the constant or register; the initial value of downward counters is the value of the constant or register, the target value is 0.
   d. When using the register type counters, if the register value changes, the target value will be updated only when the input signal turns from OFF to ON and continue the counting. Changing target value won't make the counter restart from its initial value.
   e. If the target value happens to be smaller than current value after modifying the register value, the counter will take it as reaching the target value when the input signal turns to ON. The output signal is ON and the corresponding counter contact will also be ON.
   f. One way counter means it won't return to the initial value after reaching the target value, the counter reset component is needed to reset the value.
3. Amount: Total 256 counters for use (CT0~CT255)

Ring Counter

1. Figure

	
Upward ring counter of constant type	Upward ring counter of register type

	
Downward ring counter of constant type	Downward ring counter of register type

2. Description:
   a. When the input signal changes from OFF to ON, the counter will add (or subtract) 1 time till reaching the target value.
   b. When the counter reaches the target value, the output signal is ON and its corresponding counter contact is also ON.
   c. When the ring counter reaches the target value, it'll return to initial value +1(or -1) after next trigger then keep on counting.
   d. The initial value of upward counters is 0, the target value is the value of the constant or register; the initial value of downward counters is the value of the constant or register, the target value is 0.
   e. When using the register type counters, if the register value changes, the target value will be updated only when the input signal turns from OFF to ON and continue the counting. Changing target value won't make the counter restart from its initial value.
   f. The ring counter will restart from the initial value automatically after reaching the target value. So if the counter counts to 7 and then modify the register value to 5, the counter will "reach the target value" and "being triggered" at the same time when the input signal turns to ON. Thus the counter will return to initial value+1 (or -1) and the output signal won't be ON.
3. Amount: Total 256 counters for use (CT0~CT255)

Counter Reset

1. Figure


Counter Reset

2. Description：When the input signal changes from OFF to ON, the counter resets to the initial value. The value of '#' causes no effect to the execution.

2.2.9 Flow

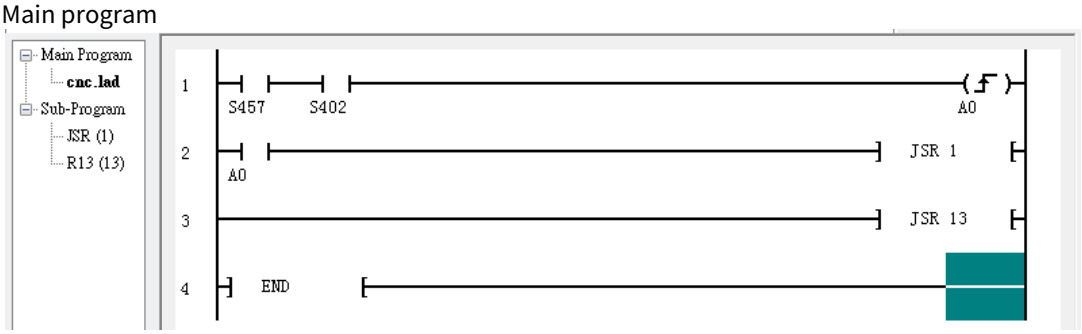
Jump Subroutine

1. Figure

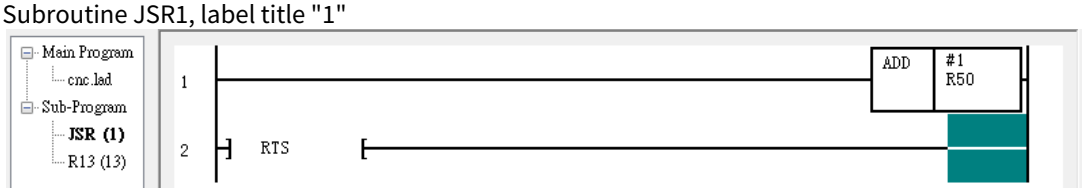

Jump to subroutine

2. Description： When the input signal is ON, executes the command of jump subroutine. The subroutine assigned by the label of JSR component will be executed; after the subroutine finish, it'll return to the main program and proceed the next command.
3. Notifications：
    a. Please organize the content and control the number of subroutines before applying. For example, use indirect addressing to reduce the subroutines with same content.
    b. Too many subroutines may lead to ladder diagram error and make JSR components not able to find the corresponding subroutines, it's the limitation of the software structure. Please reduce the number of subroutines to avoid the syntax error caused by system overload.
    c. The content of subroutine won't be updated if corresponding JSR component is not triggered.
4. Example：
    a. There are 2 subroutines in the ladder diagram. The number of times that subroutine JSR1 being executed is recorded in R50; the number of times that subroutine R13 being executed equals to the number of scans of the main program.
    b. When switching controller mode in subroutine R13, the value of R13 register will be updated after next scan.

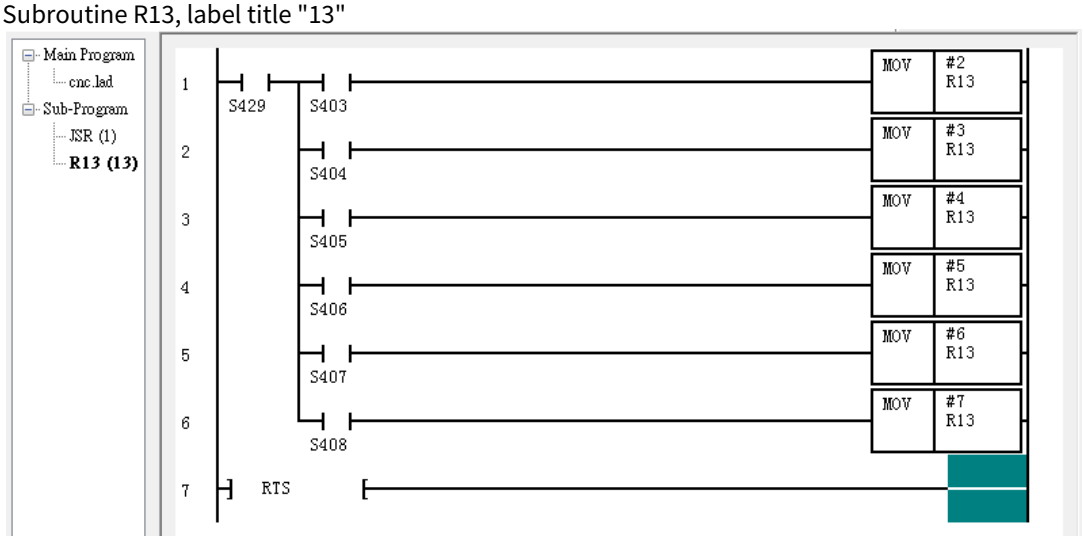
c. Main program



Subroutine JSR1, label title "1"



Subroutine R13, label title "13"



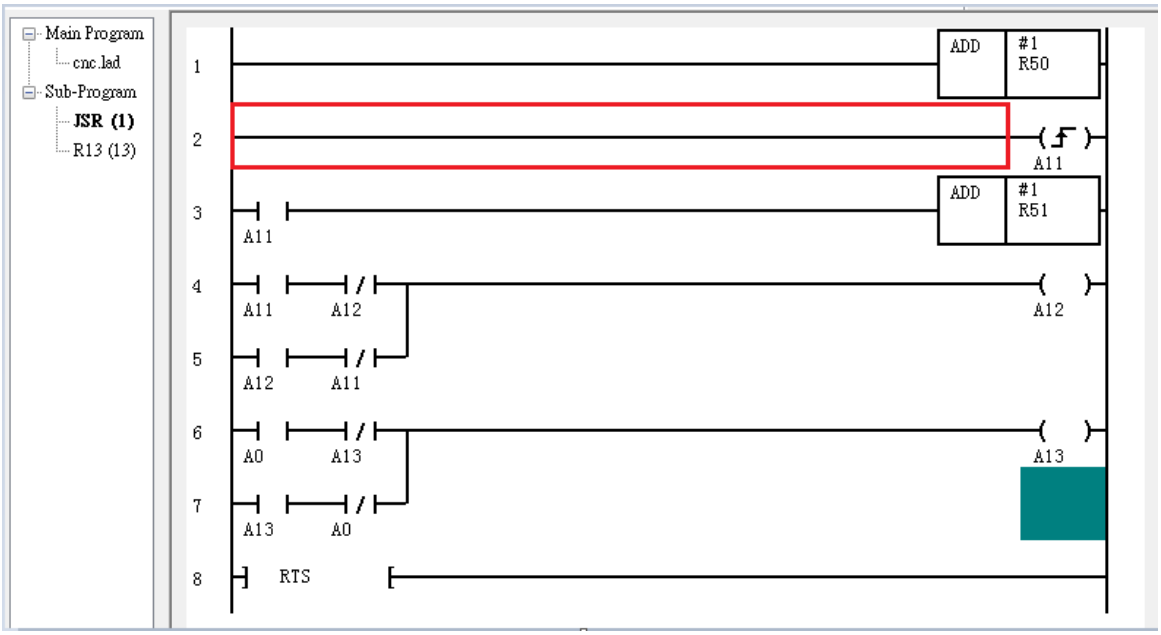
5. Special Examples

Continue the former example, modify the content of subroutine JSR1 as picture below：

a. A11 is valid only at the first execution of the subroutine JSR1, so R50 will keep adding up since JSR1 is still executed, but R51 remains 1 after the first execution.

b. A12 will remain ON after the first execution since there is nothing can be done on A11.

c. A13 can do the one-button switch action when the subroutine JSR1 is executed.

6. 

End

1. Figure


End of the program

2. Description：It's the ending command of ladder diagram and the final statement in the ladder diagram. Unable to edit under this component.

FastPLC

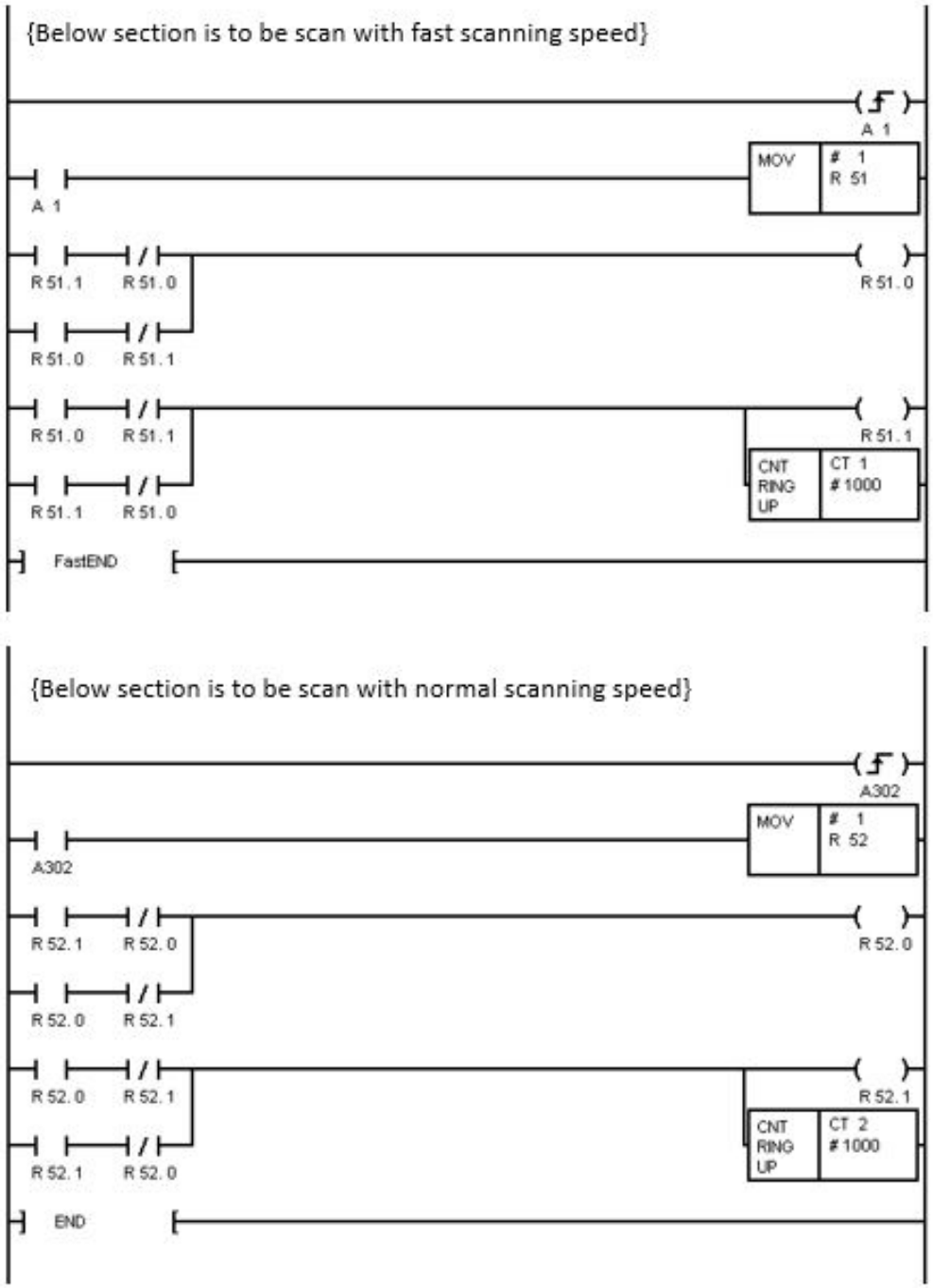
1. Figure


FastPLC

2. Description
   a. Please put the content of FastPLC at the beginning of ladder diagram and end with -]FEND[- component.
   b. When executing the ladder diagram, the content before -]FEND[- will be loaded into the fast scan area.
   c. The content between -]FEND[- and -]END[- will be loaded into the normal scan area.
   d. The scanning time of fast scan area equals to Pr3203 (interpolation time).
   e. The scanning time of normal scan area equals to Pr3204 (PLC scanning time).

f. Note：Do not put JSR component in the fast scan area.

3. Example



2.2.10 CNC

ROT

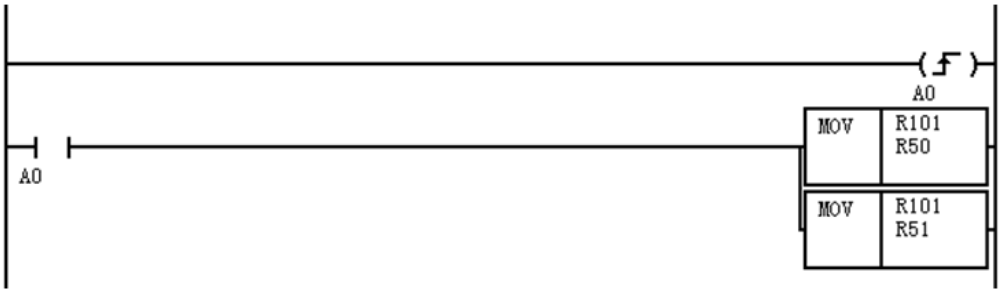
1. Figure

Tool Magazine Rotation

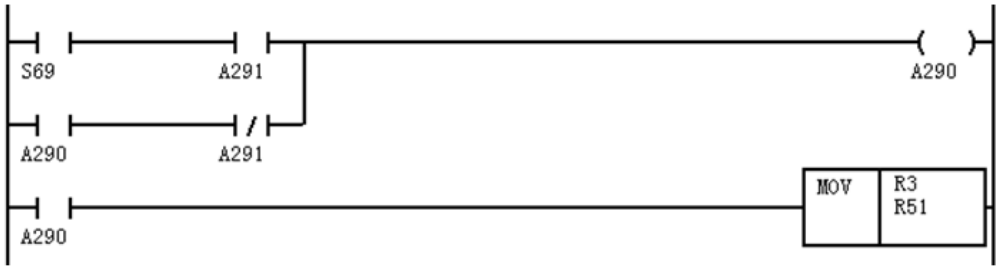
2. Description
   a. When the input signal is ON, starts to calculate the rotation direction of tool magazine, then output 1 (CW) or 0 (CCW) signal according to the calculation result. The register used are explained below：
      i. Ra - current tool number 1~n
      ii. R[a+1] - target tool number 1~n
      iii. Rb - total tool number n
   b. When this command is executed, if the current tool number exceeds the total tool number, it'll be automatically modified to the corresponding number in the reasonable range.
   c. The component decides the CW/CCW signal in the way of shortest distance. For example, the total tool number is 8, current tool number is 4 and the target tool number is 6, it'll output 0 (CCW) signal; if the target tool number is 2, it'll output 1 (CW) signal.
3. Example：If set R50 as the current tool number and R89 as the total tool number, the system will automatically set R51 as the target tool number (R[50+1]). Following is the instruction of tool changing by T code in PLC.

Contact	Description	Contact	Description
A0	Rising edge-trigger after booting	A293	Tool change direction signal
A16	Tool magazine CW rotating state	I21	Counting sensor
A17	Tool magazine CCW rotating state	R101	Spindle tool number
A290	Tool changing state	R102	Tool magazine tool number
A291	Tool change waiting state		

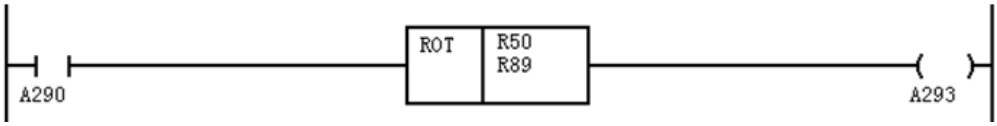
After booting, the rising edge trigger writes the spindle tool number (R101) into current tool number (R50) and target tool number (R51) for initialization.



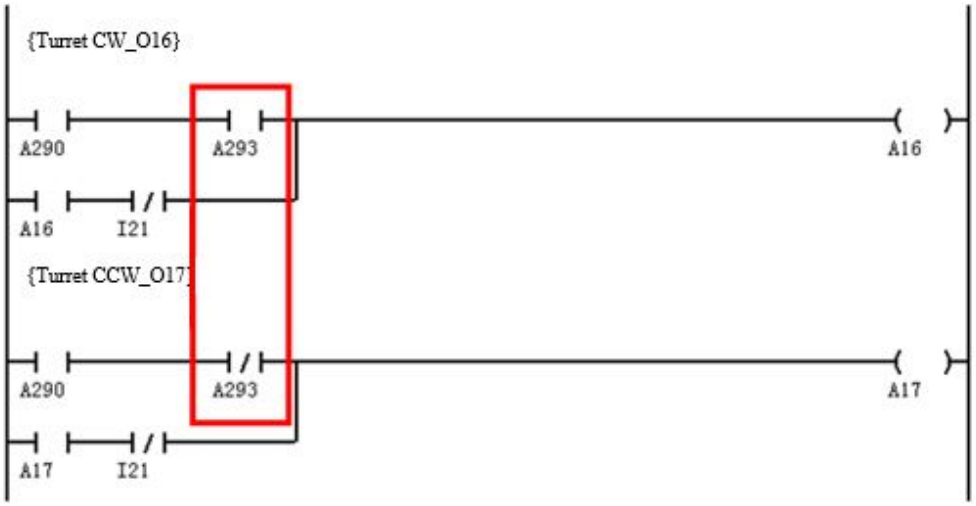
When the user applies T code (S69) to change tool, it goes into tool changing state (A290) and writes the T code tool number (R3) into target tool number (R51).



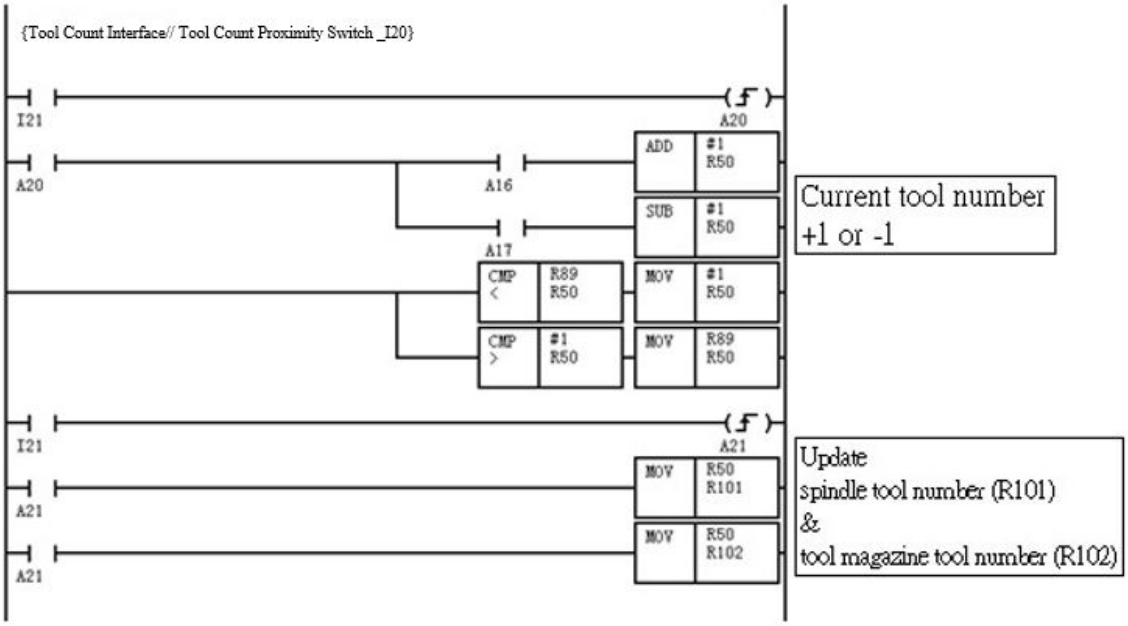
Now the **ROT** component decides the CW/CCW rotation. Output "1" for CW rotation, A293 ON; "0" for CCW rotation, A293 OFF.



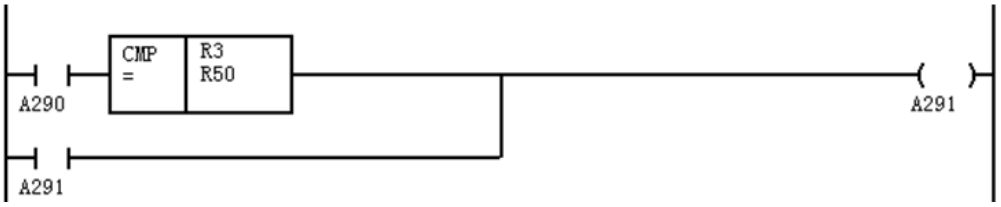
A293 decides next step：tool magazine CW rotation (A16) or CCW rotation (A17).



When receive the signal from counting sensor (I21), it means the tool magazine rotated.
The current tool number should +1 or -1, and both spindle tool number (R101) and tool magazine tool number (R102) should be updated.



The actions above repeat continuously till the current tool number (R50) equals to target tool number (R3) then goes into A291 and finish the tool changing process.

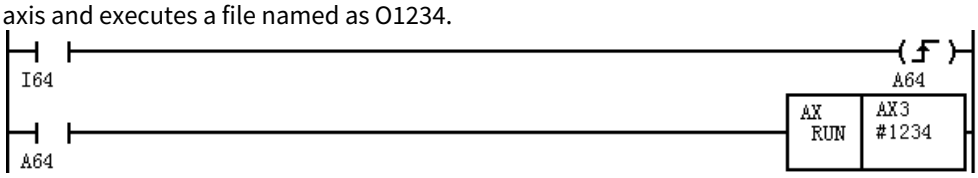


PLC Axis

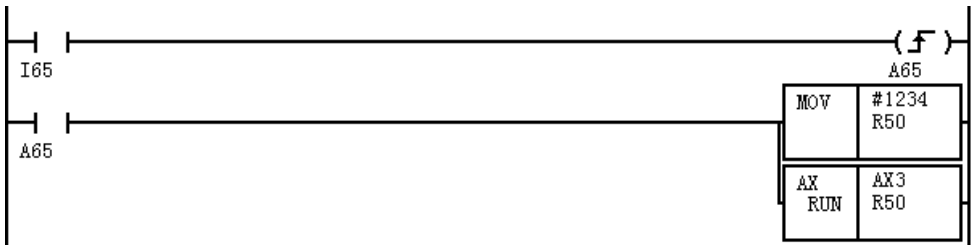
1. Figure

	
Drive PLC axis in the way of running NC programs and the filename will be assigned by a constant.	Drive PLC axis in the way of running NC programs and the filename will be assigned by a register.
	
Drive PLC axis by setting in ladder diagram	Stop PLC axis

2. Description
   a. Rn subroutine component, file number range: O0001~O9999
      - The filename is assigned by a constant. The example below assigns the 3rd axis as PLC axis and executes a file named as O1234.



      - The filename is assigned by a register. The example below assigns the 3rd axis as PLC axis and executes a file named as O1234.

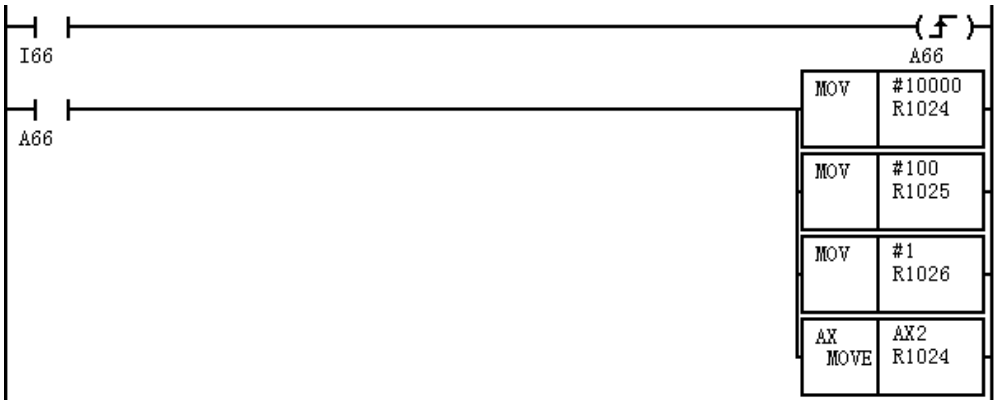


      - Subroutine of PLC axis (Onnnn)
        - The filename must start with "O", but no need to write "O" in the ladder diagram. The "nnnn" must be 4 digit natural number.
        - DOS system path: \CNC\MACRO
        - CE system path: DiskC\OpenCnc\MACRO
        - The program needs to be in MACRO form, so the content should start with **%@MACRO.**
        - Can only execute single axis movements and the axis name in the program content is always **X.**
        - All system variables about axis are invalid except X axis.
          - For example, #1301 represents the terminal coordinate of the block of PLC axis, #1302 is not defined, please do not apply in the program.

- All G code for single axis are applicable (G00、G01、G04、G90、G91、G94、G95…).
- The override of G00/G01 in PLC axis subroutine is decided by R661~R676, not R18/ R16.
- All the velocity/acceleration/jerk of G code are refer to controller parameter setups.
  - For example: G00 speed is referred to Pr461~; G00 acceleration is referred to Pr441~; G01 speed is referred to Pr621~; G01 acceleration is referred to Pr541~.
- Do not end the program with **M02, M30 or M99.**
- When executing M code in the program, please use **R629/ R630/ R681** for M code read/ M code finish/ M code content, which is equivalent for S29/ C38/ R1.
- Tool setup functions such as **G10 L10/L11/L12/L13/L14** are not supported.

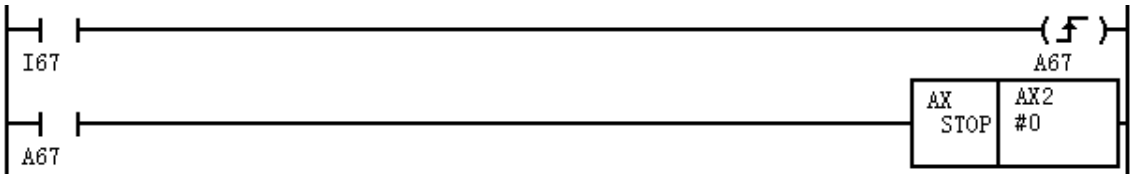
b. MoveAx component, moving in the assigned speed, distance and G90/G91 mode

- MoveAx component needs 4 continuous register to set up the action content, below are the definitions:
  R(base address+0)：moving distance, unit: BLU
  R(base address+1)：moving speed, unit: mm/min
  R base address+2)：absolute/increment mode selection, 0: increment mode; 1: absolute mode
  R(base address+3)：no function for now, saved for further expansion
- MoveAx component should be located under these registers.
- Example: The 2nd axis moving 10mm in the speed of 100 mm/min in absolute mode, base address is 1024.



- The acceleration of MoveAx is referred to Pr441~, and the override is decided by R18.

3. StpAx component: Stop all actions of PLC axis. Example: Stop the moving 2nd axis. The value of "#" causes no effect to the execution.



4. The executing PLC axis actions can also be stopped by Reset button, but it affects other actions such as machining process.
5. During the machining, if PLC axis action is executed while the axis is still moving, it'll cause command overlapping and lead to wrong moving path.
6. When executing PLC axis, please apply rising edge-trigger to avoid the same action being triggered multiple times.
7. PLC axis supports MPG simulation function, please refer to **R601** for further details.

8. Don't execute PLC axis function in JOG/ INJOG/ MPG/ HOME mode. If necessary, Please take enough safeguard because It may cause command overlapping and lead to wrong moving path.

Keyboard Scan

1. Figure

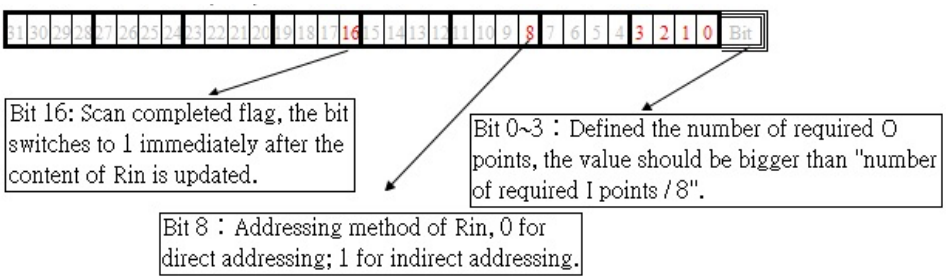
Keyboard Scan

2. Description
   a. Activate PLC keyboard scan when the component is triggered, the input data is read by DB-25PIN of PIO5 (I/O board) on the controller; the input data is stored from Low Bit to High Bit.
   b. From the PIN definition of DB-25PIN of PIO5 can know that one O point is able to scan in 8 I points. If there are five O points, then it'll be able to input utmost 40 I points. Below is the data input order:
      i. Bit 0~31 of Rin(base address+0) corresponds to I point 0~31（Rin means register for input data）
      ii. Bit 0~7 of Rin(base address+1) corresponds to I point 32~39
   c. At least 2 continuous registers will be used in PLC keyboard scan, R(base address+0) content is organized as below, the way of using R(base address+1) is different according to the addressing method.

R(base address+0):

Bit 0~3: Defined the number of required O points, the value should be bigger than "number of required I points/8".
Bit 8: Addressing method of Rin, 0 for direct addressing; 1 for indirect addressing.
Bit 16: Scan completed flag, the bit switches to 1 immediately after the content of Rin is updated.

Function definition of R(base address+0):



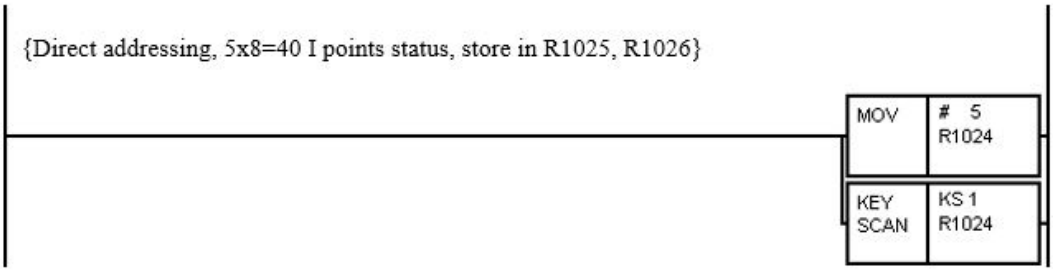
Rin is the input data register, which is the register assigned to store the data, below is the definition:

For direct addressing, the scanning content is saved in R(base address+1) ~ R (base address+4);
For indirect addressing, Rin is assigned by R(base address+1).

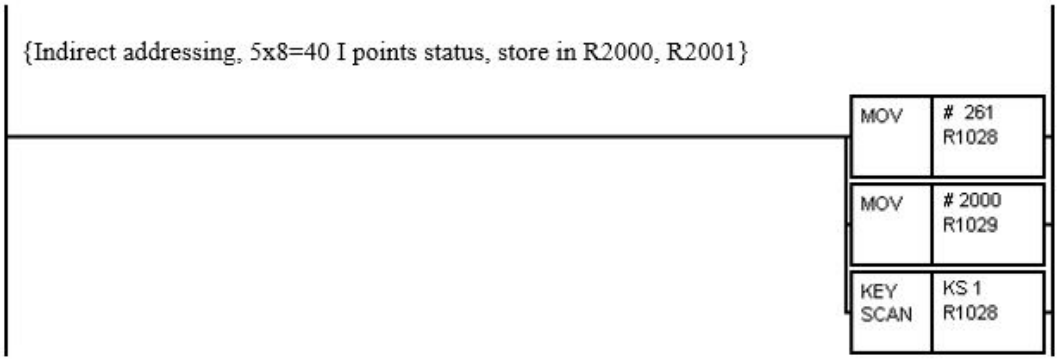
The table below is the schematic diagram of Rin saving I point input:

31	30	29	28	27	26	25	24	23	1	1	1	19	18	17	16	15	14	13	12	1	1	1	8	7	6
31	30	29	28	27	26	25	24	23	On	On	On	19	18	17	16	15	14	13	12	On	On	On	8	7	6

3. Only supports 1 set of KS component and is only valid when applying PIO5 hardware.
4. Example
   a. Direct Addressing: Using five O points as control signal, able to scan in 40 I points. The content will be saved in R1025 & R1026.
   R(base address+0)= R1024, Rin=R(base address+1)=R1025 is the first input data register, R(base address+2)=R1026 is the second input data register.



   b. Indirect Addressing: Using five O points as control signal, able to scan in 40 I points. The content will be saved in R2000 & R2001.
   R(base address+0)= R1028, R1028=261 means bit 8=1 is indirect addressing; bit 2=bit 0=1 means using 5 O points.
   R(base address+1)=R1029=2000 means Rin=R2000 is the first input data register; R2001 is the second.



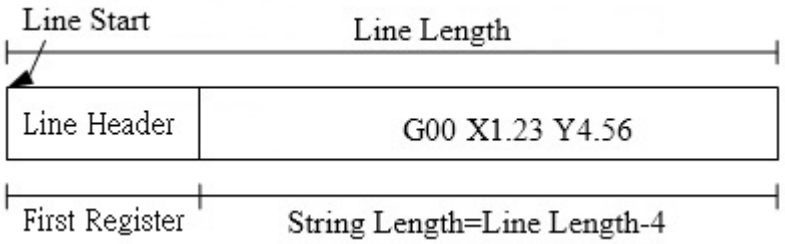
PLC Coordinate Teaching System

1. Figure

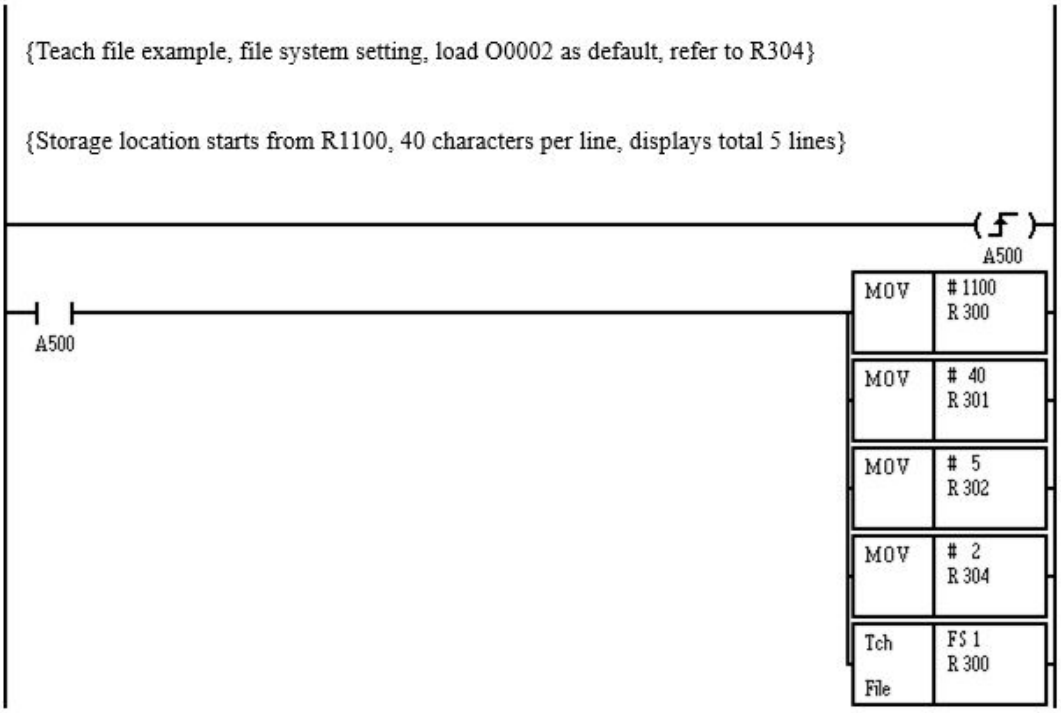
PLC Coordinate Teaching

2. Description
    a. Activates PLC coordinate teaching system when the input signal switches form OFF to On, the system will stay effective after activated till machine power off.
    b. Users can build file system based on registers through ladder diagram.
    The PLC coordinate teaching system can execute action teaching such as G00 and G01, then save the teaching result for users to do file managements through ladder diagram.
    c. PLC coordinate teaching system component requires 8 continuous registers, the functions are defined below:
    R(base address+0): [In], address of the initial register of data showing area
    R(base address+1): [In], number of bits occupied by each row in data showing area
    R(base address+2): [In], number of rows showing at once in data showing area
    R(base address+3): [In], address of the register for control command (teaching, delete, up, down etc.)
    Bit 0: G00 teaching, takes the current machine coordinate as terminal point when switches to On.
    Bit 1: G01 teaching, takes the current machine coordinate as terminal point when switches to On.
    Bit 2: G02/G03 teaching, reserved function for now, no function.
    Bit 3~5: Reserved
    Bit 6: Delete line, delete the line where the cursor at when switches to On.
    Bit 7: Save file, save the current editing file into the disk when switches to On.
    Bit 8: Move cursor to the last line when switches to On.
    Bit 9: Move cursor to the next line when switches to On.
    Bit 10: Move cursor to the previous page when switches to On.
    Bit 11: Move cursor to the next page when switches to On.
    Bit 12~31: Reserved
    R(base address+4): [In], the register used to assign the filename of teaching file (NC file), the filename should start with "O", range: 1 ~ 8999.
    R(base address+5): [Out], the filename of current teaching file (NC file) of coordinate teaching system
    R(base address+6): Reserved
    R(base address+7): Reserved
    [In] means the value filled in by users, [Out] means responding from the ladder diagram
3. Currently only supports 1 PLC coordinate teaching system.

4. Remarks：Format of data showing area
   In the line length set by users, the first register (4 bytes) is used to save the line data, the second register then starts being used to save the string data (4 bytes per register).
   Format of the first register (Line Header) is mentioned below:
   Bit 0~15: Save the line number of the line, which is the line number in the document (count from line 0).
   Bit 16: On means it's the cursor line; Off means it's not.
   Bit 17~31: Reserved

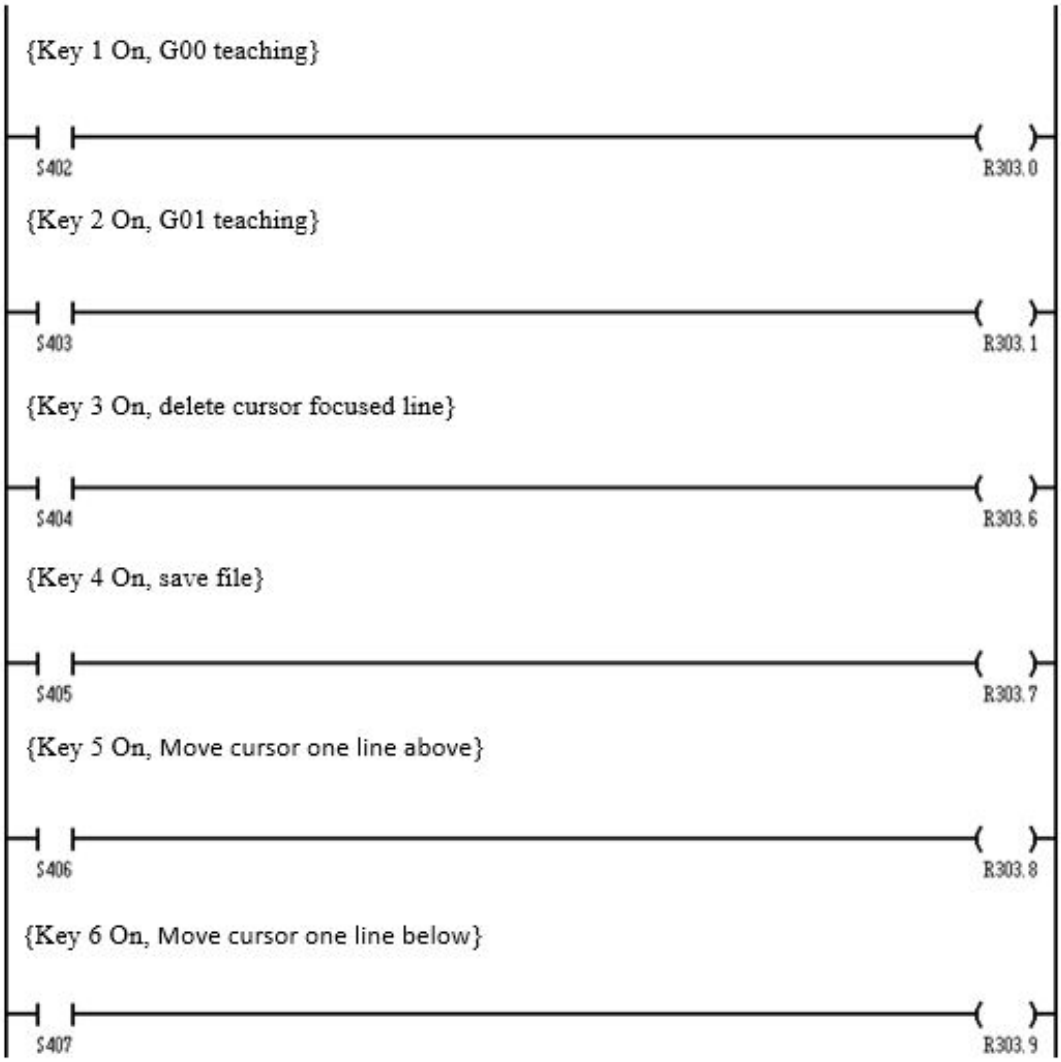


5. Example
   a. Setting PLC coordinate teaching component
      The assigned data showing area starts from R1100 (base address=300), 40 bytes each line with total 5 lines.
      Set O0002 to be the loaded NC file during initialization, the process only needs to be executed once.



b. Choose the teaching file (NC file)
Put the filename of the file you want to teach into R(base address+4), it's R304 in this example.
As shown below, the NC file loaded is O0003.

c. Teaching control action
Set up R(base address+3) according to the required actions and let the corresponding bit be On, it's R303 in this example.
As shown in the picture below, it's able to do G00 teaching, G01 teaching , delete line, save file, cursor moving actions in order.



2.2.11 RS232

RS232 Command

1. Figure

Data transmit and receive

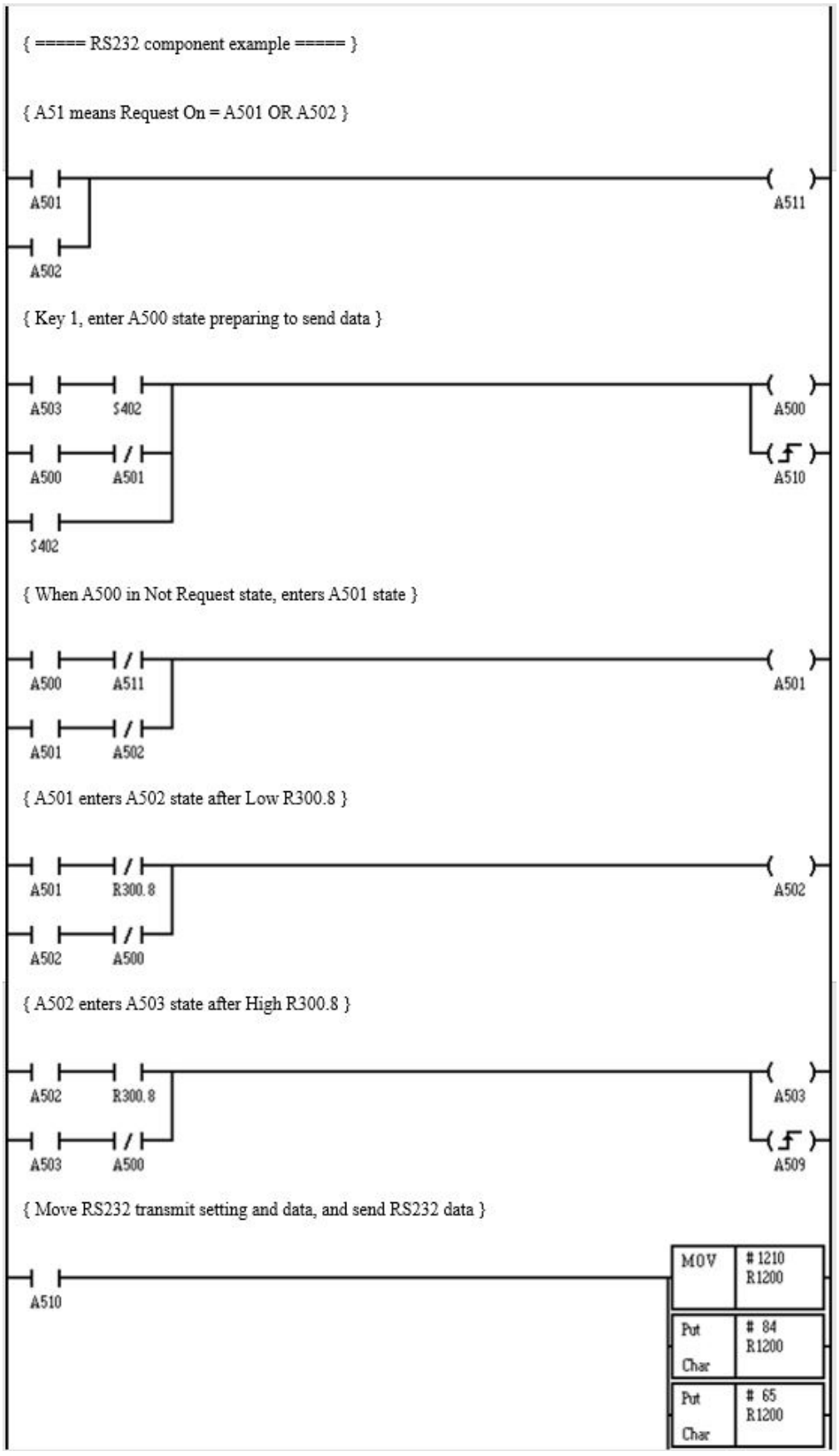
2. Description
   a. Activate RS232 transmission command when the input signal switches from OFF to ON. Starting from the initial register, the data in several registers which are assigned will be sent through RS232.
   If there's any return value, RS232 component will start filling in the assigned register with the received data in order and record the number of bytes received, then return to waiting state.
   b. This component needs 8 continuous registers to set up the action content, below are the definitions:
   R(base address+0): [Transmit/Receive], RS232 control register.
   Bit 0: Transmission waiting control bit. If the bit is On, it means the component requires return value after transmitted the data to proceed the next transmission; if the bit is Off, it means the component can proceed the next data transmission without waiting for return value.
   Bit 1 ~ 7: Reserved
   Bit 8: Data receiving bit, should be On under normal situation. When the input signal switches to On, this bit switches to Off first then transmit the data, then changes back to On when return value is not required or after the return value is received.
   Bit 9: If the bit is On, it means there's an error during RS232 transmission; if the bit is off then means there's no error during the transmission.
   Bit 10: If the bit is On, it means there's a Time Out error during RS232 reception; if the bit is off then means there's no error during the reception.
   Bit 11: If the bit is On, it means the register value of RS232 transmission is wrong; if the bit is off then means the value is correct.
   Bit 12: If the bit is On, it means there's I/O error during RS232 transmission/reception.
   Bit 13 ~ 31: Reserved
   R(base address+1): [User fill-in], the address of the initial register for transmitting
   R(base address+2): [User fill-in], the number of bytes to transmit (unit: R)
   R(base address+3): [User fill-in], the address of the initial register for saving return value
   R(base address+4): [Transmit/Receive fill-in], assign the largest number of receiving bytes, RS232 will fill in the number of received bytes in real.
   R(base address+5): [User fill-in], receive the ending bit (terminator)
   R(base address+6): Reserved
   R(base address+7): Reserved
   c. Hardware Com Port and the corresponding parameter:

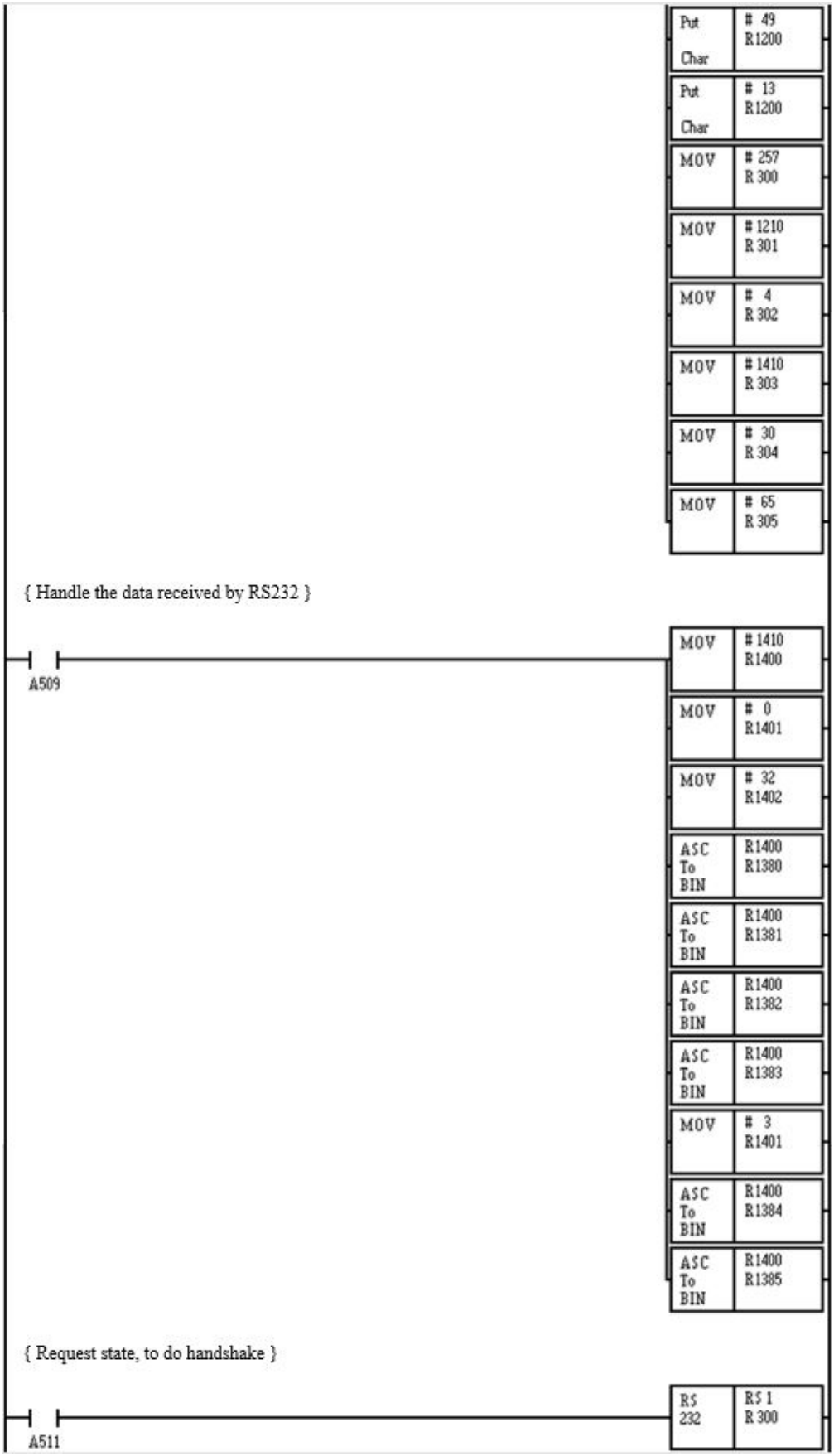
	SUPER/10/20/200	eHMC/3/6/11/21
RS232（Pr3921~Pr3930）	COM1	-
RS422（Pr3941~Pr3950）	COM2	-
RS485（Pr3961~Pr3970）	COM3	COM2

3. RS232 component only supports single Com Port. For example: 200 series controller, if already using RS232 component to transmit data through COM1, you can't transmit data through other Com Port with RS232 component.
4. Example
    a. Control the transmission/reception of RS232 with 4 states A500~A503, below are introduction to actions of the 4 states. Button 1 (S402) can lead the process into A500.
    A500：
    Prepare for RS232 data transmit. Triggers a positive edge signal A510 to prepare for data transmission than goes into A501 immediately.
    A501：
    The Request state of RS232, will first set the bit 8 of the control register Off. Then goes into state A502.
    A502：
    The Request state of RS232, if bit 0 of the control register is On, the data will be transmitted through RS232 and await for the return value; if bit 0 is Off then won't wait.
    A503：
    Dealing with return value. Triggers a positive edge signal A509 to deal with the return value. Enters A500 again after received the signal of Button 1 (S402).

    b. Prepare for data transmit
    For data transmitting from RS232, input 84, 65, 49, 13 into 4 registers starting from R1210, the 4 numbers are ASCII code for "TA1\n".
    Set the address of the initial register for the transmission as R1210, so fill in R301 with 1210.
    Set transmission length to 4 and fill in R302.
    Set the address of the initial register for the return value as R1410, so fill in R303 with 1410.
    Set maximum reception length as 30 and fill in R304, then assign "13" as the terminator.

    c. Dealing with return value
    Assume the return value to be "EV 301 0 1 23.827 159.447\n" and start filling in from R1410. So fill in R1400 with 1410, which is the initial address of the register.
    Set the number of decimal places to 0 (means integer) and fill in R1401.
    Set the value splitter into blank, the ASCII code is 32, and fill in R1402.
    Execute the "character to number" conversion for the first time, it'll transform "EV" into numbers and fill in R1380. The conversion result is 0 since they're English alphabets.
    The second execution will convert "301" into 301 and fill in R1381; "0" into 0 and fill in R1382; "1" into 1 and fill in R1383.
    After, set the number of decimal places to 3 then convert "23.827" into 23827 and fill in R1384; "159.447" into 159447 and fill in R1385.


{ ===== RS232 component example ===== }
{ A51 means Request On = A501 OR A502 }
A501
A502
A511
{ Key 1, enter A500 state preparing to send data }
A503
S402
A500
A501
S402
A500
A510
{ When A500 in Not Request state, enters A501 state }
A500
A511
A501
A501
A502
{ A501 enters A502 state after Low R300.8 }
A501
R300.8
A502
A502
A500
{ A502 enters A503 state after High R300.8 }
A502
R300.8
A503
A503
A500
A509
{ Move RS232 transmit setting and data, and send RS232 data }
A510
MOV
1210
R1200
Put
Char
84
R1200
Put
Char
65
R1200


Put Char # 49 R1200
Put Char # 13 R1200
MOV # 257 R 300
MOV # 1210 R 301
MOV # 4 R 302
MOV # 1410 R 303
MOV # 30 R 304
MOV # 65 R 305
{ Handle the data received by RS232 }
A509
MOV # 1410 R1400
MOV # 0 R1401
MOV # 32 R1402
ASC To BIN R1400 R1380
ASC To BIN R1400 R1381
ASC To BIN R1400 R1382
ASC To BIN R1400 R1383
MOV # 3 R1401
ASC To BIN R1400 R1384
ASC To BIN R1400 R1385
{ Request state, to do handshake }
A511
RS 232 RS 1 R 300

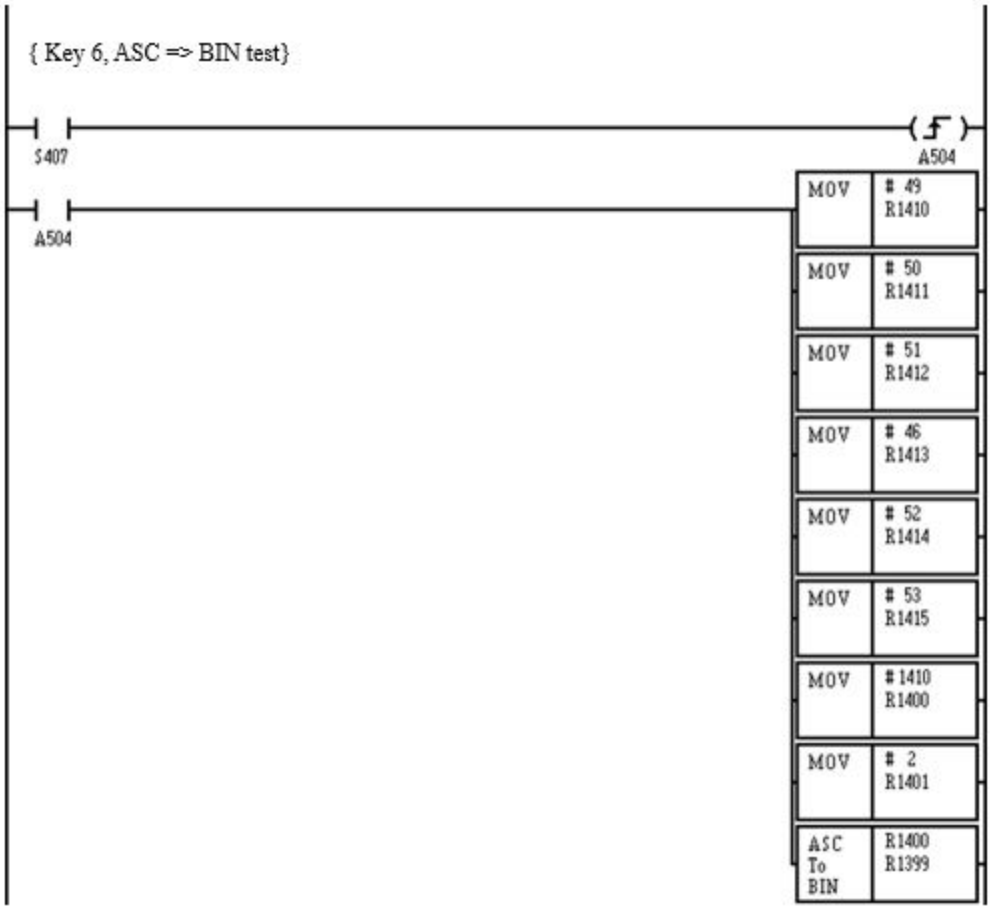
ASCII Code to Numbers

1. Figure

ASCII Code to Numbers

2. Description
   a. When the input signal is ON, it'll convert the register values of an continuous assigned area into numbers with the ASCII code table, then fill in to target registers after shifting the numbers according to the assigned number of decimal places. For example, if the conversion result is 12.345 with number of decimal places set to be 3, the target register will be filled in with 12345. （1000 times larger）
   b. The registers for this component are defined below:
   Source Register R1(base address+0): Saves the initial register address of the converting ASCII code.
   Source Register R1(base address+1): Set the number of decimal places and shift according to the set value. (amplification ratio).
   Source Register R1(base address+2): Splitter used to separate the values in the continuous registers.
   Source Register R1(base address+3): Reserved
   Target Register R2(base address+0): The initial register address to be filled in after conversion.

c. Example
The values stored in registers from R1410 is 49 (ASCII code for "1"), 50 (ASCII code for "2"), 51 (ASCII code for "3"), 46 (ASCII code for "."), 52 (ASCII code for "4"), 53 (ASCII code for "5"). The register address for storage is 1410 set by the source register R1(base address 1400) and R1401 is used to save the number of decimal places, 2. The value after conversion is 123.45 and becomes 12345 after x100, then saves into the target register R1399. The register address set by source register R1 will change to 1416, which is the address available for next conversion.

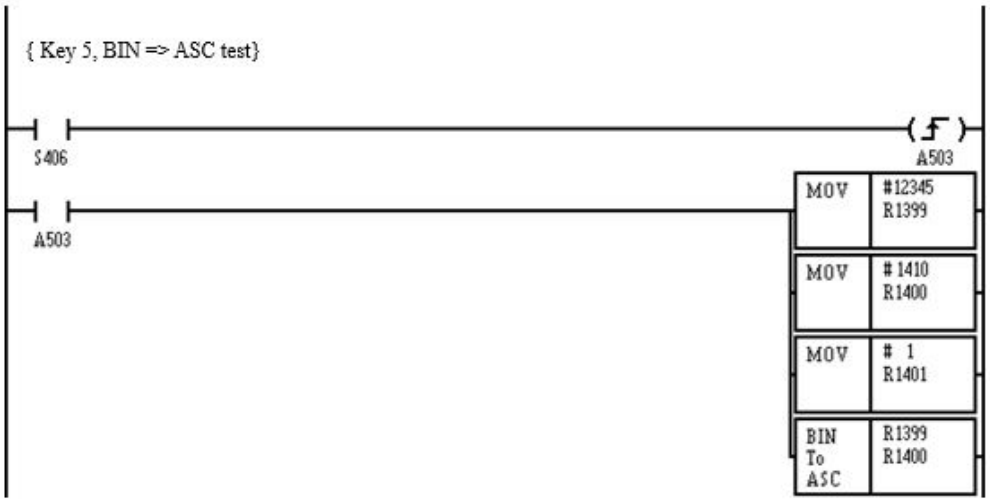


Numbers to ASCII Code

1. Figure

Numbers to ASCII code

2. Description
   a. When the input signal is ON, it'll read the value stored in the source register and converts the value shifted according to the assigned number of decimal places into ASCII code digit by digit. Then fill in from the initial target register address.
   For example, the value from source register is 12345 with number of decimal places set to be 3, it'll be converted into 12.345. From the initial address of the assigned target register, ASCII '1', ASCII '2', ASCII '.', ASCII '3', ASCII '4', ASCII '5' will be filled in.
   Please note that this component is not a positive edge-trigger component.
   b. The registers this component requires are defined below:
   Source Register R1: Saves the register address which stores the converting numbers.
   Target Register R2: The initial register address to be filled in after converted to ASCII code.
   Target Register R2(base address+1): Set the number of decimal places and shift according to the set value. (minification ratio).
   Target Register R2(base address+2): Reserved
   Target Register R2(base address+3): Reserved
   c. Example
   Set the source register (R1399) to be 12345; set R1400 as 1410 to be the initial register address for saving ASCII code; set the number of decimal places to be 1 with R1401.
   After the execution, 12345 will be reduced by 10 times to 1234.5 and starts filling in from R1410 in the order of 49 (ASCII code of "1"), 50 (ASCII ode of "2"), 51 (ASCII code of "3"), 52 (的ASCII code of "4"), 46 (ASCII code of "."), 53 (ASCII code of "5"). Then changes the initial register address set by R1400 from 1410 to 1416, which is the address available for next conversion.

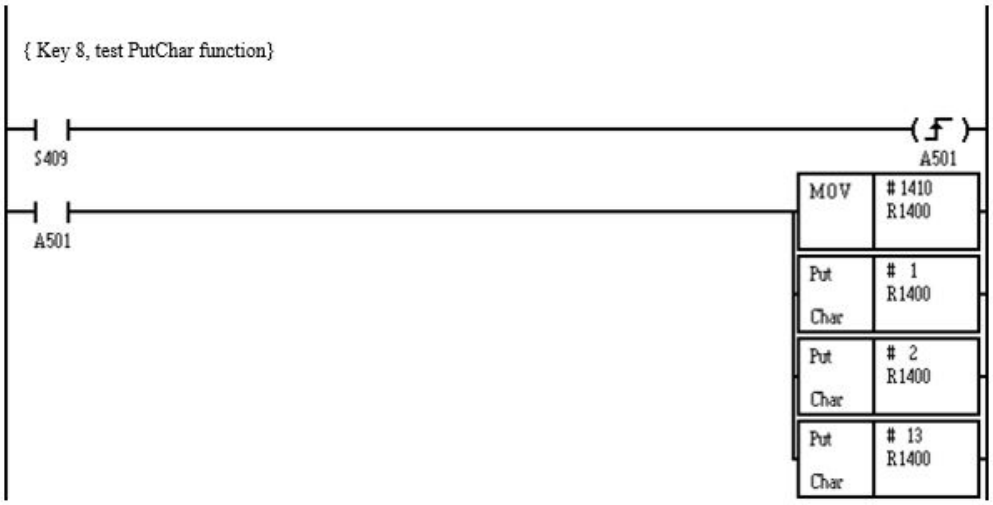


Input Value to Register

1. Figure

Input Value to Assigned Register

2. Description
   a. When the input signal is ON, it'll write the input value into target register, after the value is wrote in, the target register address will move to the next.
   b. The component is not a positive edge-trigger component.
   c. The registers this component requires are defined below:
   Source Register R1: Saves the register address which stores the inputting value.
   Target Register R2: Saves the register address which will be written in.
3. Example
   Source register R1400, target register R1410.
   After pressing button 8, write "the register address 1410 which will be written in" to R1400 first, then write in 1, 2, 13 in order. The 3 numbers will be fill in R1410, R1411, R1412 respectively.
   After the writing action, value of R1400 will be changed to 1413, which is the next register address able to write in.

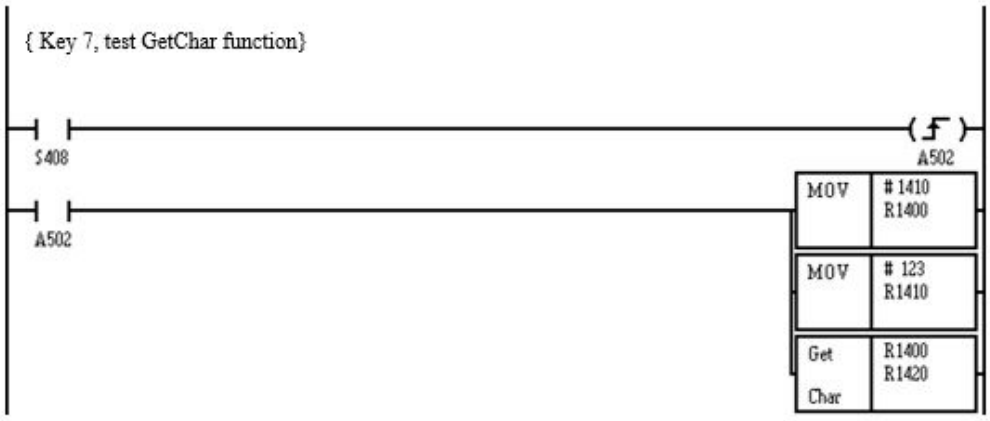


Read Value from Register

1. Figure

Read Value from Assigned Register

2. Description
   a. When the input signal is ON, it'll read the value stored in the register address pointed by source register and write in to the target register, then the register address pointed by source register will move to the next.
   b. This component is not a positive edge-trigger component
   c. The registers this component requires are defined below:
   Source Register R1: Saves the register address which stores the value be read.
   Target Register R2: Saves the register address which stores the read value.
3. Example
   Read the value stored in the register address pointed by R1400 and write in to R1420.
   After pressing button 7, the register address 1410 will be written in to R1400 first. Then fill in R1410 with value 123, after 123 is read, it'll be fill in R1420.
   After reading, the value of R1400 will point to R1411, which is the next register address available to be read.



Modbus Reading Command

1. Figure

Read multiple data from specific address of the assigned device

2. Description
   a. Available station ID for the assigned device：0~255.
   b. MODR is the MODBUS RTU16 reading component, it's able to read consecutive 1~15 data.
   c. MODBUS RTU uses the 8-bit format, corresponding to the number of bits of the COM Port setup (Pr3942 or Pr3962).
   d. The system will show the transmission state of Modbus with R608.0/.10/.31 (ref: Appendix 4-a), the error will be shown at R5039 if occurred during the transmission (ref: Appendix 4-c).
   e. Modbus COM Port is the COM Port used to set up the physical wiring, if the hardware is SUPER/10/20/200, the COM Port would be 3 (R5040=3); if it's eHMC/3/6/11/21, the COM Port would be 2 (R5040=2).
   f. Enable Modbus TCP if set R5040 to be 9, it'll be able to transmit the data through Internet (LAN), the regulations are mostly the same as Modbus RS485, below are the differences:
      i. The controller as Client (Master) and the device as Server(Slave) must be in the same domain.
      ii. The station ID of Server (Slave) should be the same as the 4th code of IP. For example, if the IP is 192.168.1.5, the station ID should set to be 5.
      iii. Able to connect with at most 2 devices as Server (Slave).
      iv. The communication timeout can be set with Pr3971.
   g. If need to change the communication protocol immediately, please apply R5030 (ref: Appendix 4-b). If R5030 is not assigned or R5030.0 not activated, please set up the standard communication protocol parameter according to the COM Port number.
   For example: If using COM2, Pr3941~Pr3950 should be set up; if using COM3, Pr3961~Pr3970 should be set up.

h. The component contents need to be set up:
Initial Reading Address：Set up the data address of external devices (Slave) to be read
For example：Set the initial reading address to be 2020 means the data address of external device should be written in R2020. Assume the data address is $0801_{(16)}$, then make R2020=2049 (usually the address of external devices will be shown in hexadecimal form, please convert to decimal form by yourself, $0801_{(16)}=2049_{(10)}$).
Initial Returning Address：The register address to store the data read from external devices to the controller (Master).
For example：If the initial returning address is set as 2021 and the data read from external devices should be filled in R60, it makes R2021=60.
Device Address：The station ID of external devices (Slave)
For example：If set the device address to be 1, it means the station ID of the external device is 1 (the station ID of external devices can be set by its own parameters).
Length of reading data：The number of data read from external devices (Slave)
For example：Set the length of reading data as 2 means the number of data returning is 2. It'll return the data of the address assigned by initial returning address (0801) + the data of the next address (0802).

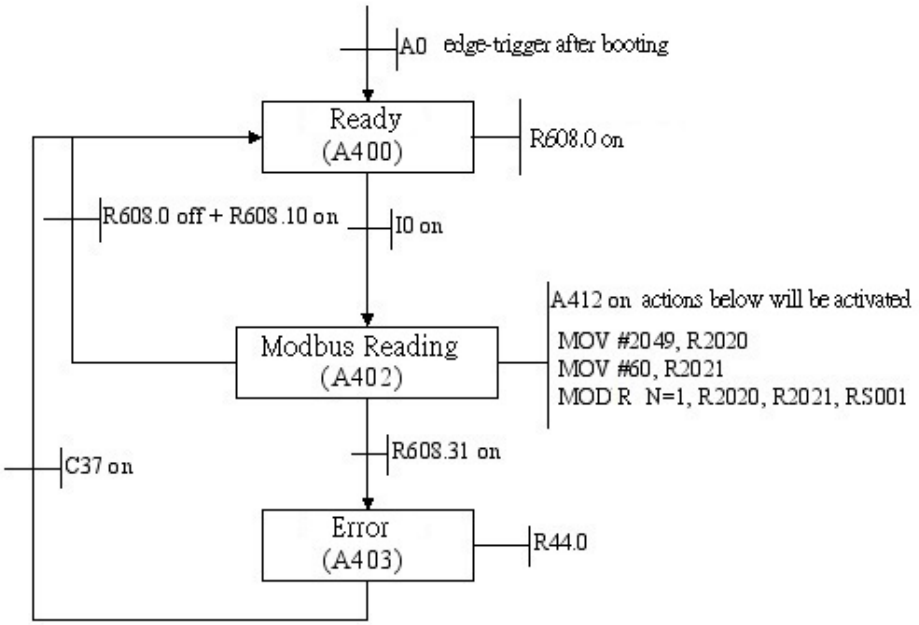
i. All the address are filled in with decimal form in the ladder diagram.
j. Please refer to the Appendix for all the other details.

3. Example:
   a. Required Specifications
   Applying eHMC controller (using COM2) for Modbus data reading communication (station ID=001, data address=0801H) with address 0801 of external device. The initial reading address of MODR component is R2020 (R2020=2049); the initial returning address is R2021=60, which means the initial address for data storage is R60. Reading value from address 0801 of external device when I0 is activated, if the signal is received properly, then it's able to receive the correct value at the assigned register address; if transmission error occurred, the transmission communication will be paused immediately and alarms will be sent, it'll return to communication ready state after pressing the Reset button.
   b. State Organize
      i. A400 Communication Ready State：
         - Enter A400 state after rising edge-triggered after booting, Reset, or after the previous data transmission is confirmed to be completed.
         - A400 should keep activating R608.0
         - The data transmission completion can be confirmed with R608, when R608.0 Off and R608.10 On means the previous data is successfully transmitted (ref: Appendix 4-a).

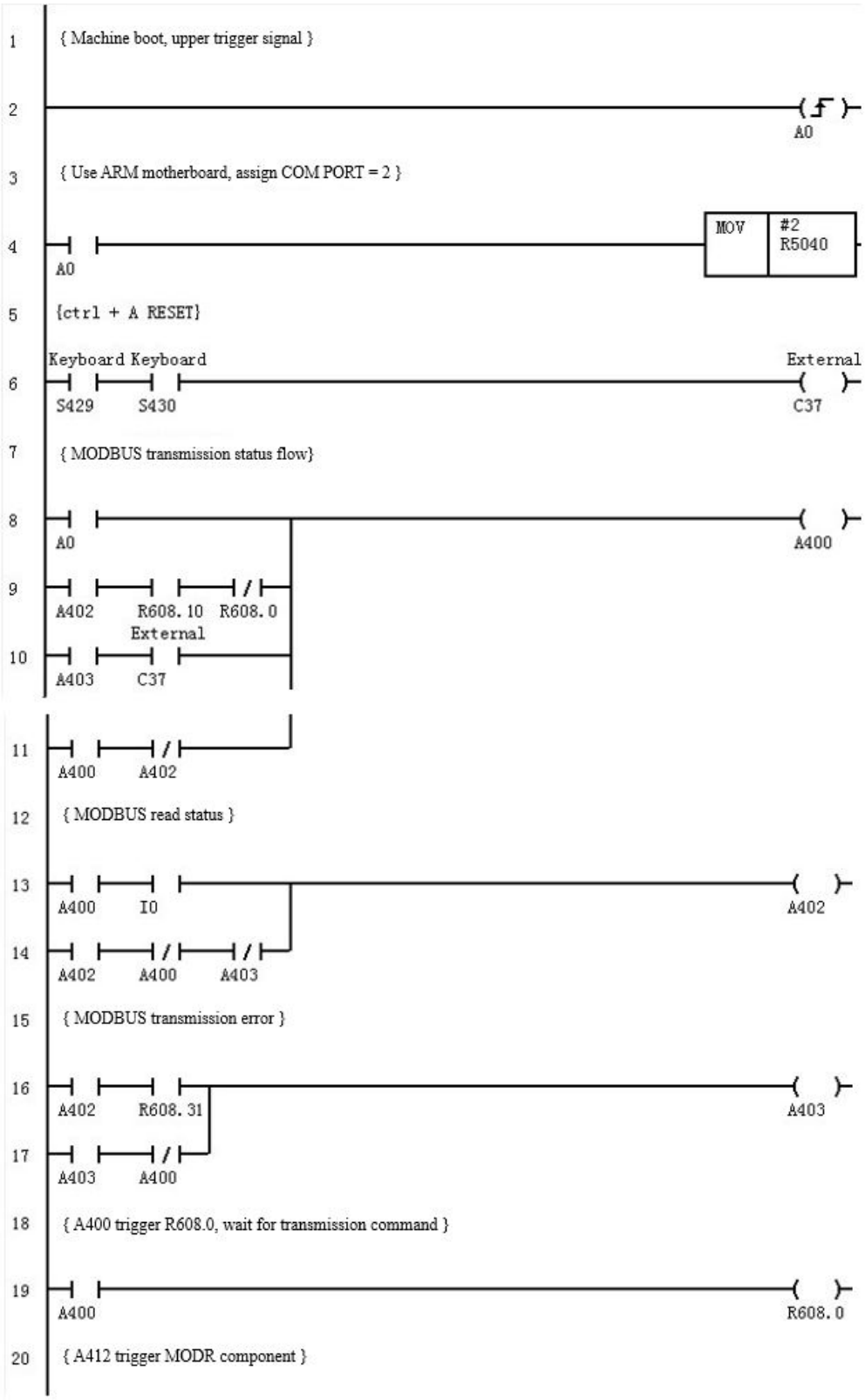
      ii. A402 Modbus Reading State：
         - In A400 state, it'll enter A402 state when I0 is activated.
         - A402 will activate A412 with rising edge-trigger, trigger the MODR component once to run the Modbus reading action.

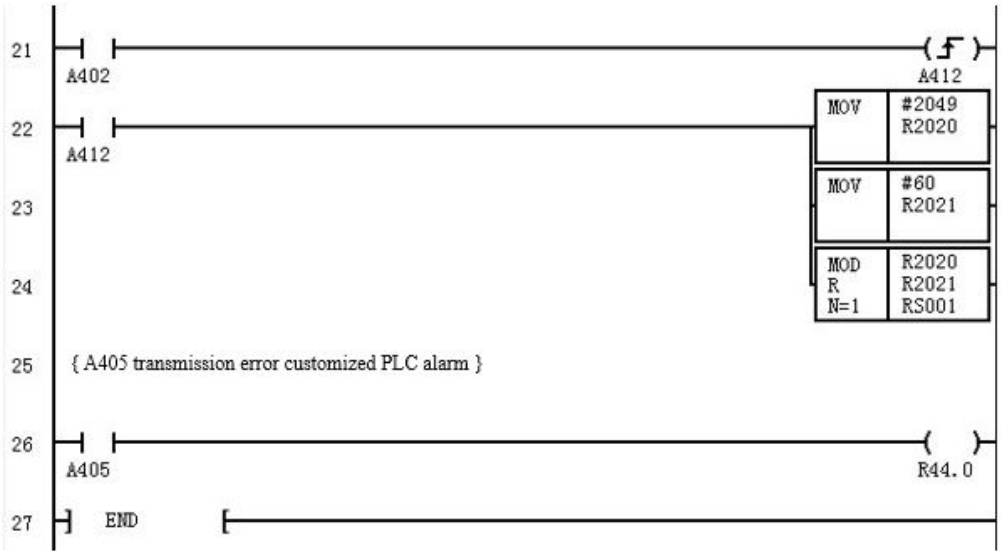
      iii. A403 Transmission Error State：
         - In A402 state, if unable to complete the Modbus action successfully, the controller will activate R608.31 automatically, which means there's a transmission handshake error. Then enters A403 transmission error state and trigger alarm R44.0 to announce the transmission error (the error code will be shown at R5039, ref: Appendix 4-c).
         - If triggers Reset in A403 state, it'll return to A400 state.

c. MSD



d. Ladder Diagram Example


1
{ Machine boot, upper trigger signal }
2
A0
3
{ Use ARM motherboard, assign COM PORT = 2 }
4
A0
MOV
#2
R5040
5
{ctrl + A RESET}
Keyboard Keyboard
External
6
S429
S430
C37
7
{ MODBUS transmission status flow}
8
A0
A400
9
A402
R608.10
R608.0
External
10
A403
C37
11
A400
A402
12
{ MODBUS read status }
13
A400
I0
A402
14
A402
A400
A403
15
{ MODBUS transmission error }
16
A402
R608.31
A403
17
A403
A400
18
{ A400 trigger R608.0, wait for transmission command }
19
A400
R608.0
20
{ A412 trigger MODR component }



e. IOCSAR

No.	Description	Note
I0	The I point triggering reading actions	
C37	Reset Button	Kernel Definition
S429	Keyboard Button: Ctrl	
S430	Keyboard Button: A	
A0	Rising edge-triggered after booting	
A400	Communication Ready State	
A402	Modbus Reading State	
A403	Transmission Error State	
A412	Reading action triggering flag	
R44.0	Modbus transmission error alarm	
R608.0	Data transmission start	Kernel Definition

No.	Description	Note
R608.10	Transmission completed with correct data	Kernel Definition
R608.31	Transmission Error	Kernel Definition
R2020	Data address to be read of external devices	
R2021	Storage address of data read from external devices to controller	
R5040	Assigned COM Port for Modbus communication	

4. Appendix
    a. Register about Modbus communication

Flag	Function	Reading/ Writing Rule
R608.0	Ready State, ready for data transmission. When R608.0 is On, R608.10, R608.31 and R5039 will be cleared automatically.	W
R608.1	Transmitting	R
R608.2	Waiting for reception	R
R608.3	Receiving	R
R608.10	Reception Complete	R
R608.31	Handshake Error	R
R5030	Communication Protocol Setup	W

R5039	Error Code	R
R5040	Set up COM Port number for Modbus communication	W

b. R5030 Communication Protocol Setup

i. When bit 0 of R5030 is set to be 0 as non-activated, the Modbus component will refer to Pr39xx during communication.
For example: When using COM2 for Modbus communication, it'll use Pr3941~Pr3950; if using COM3, then it'll be Pr3961~Pr3970.

Bit Number	**Function**	**0（Off）**	**1（On）**
b_0	To activate communication protocol setup	Not Activated	Activated
b_1	Data Length	7 bit	8 ibt
b_2 b_3	Parity Bit	None：00 Odd：01 Even：11	
b_4	Stop Bit	1 bit	2 bit
$b_8b_7b_6b_5$	Baud Rate	2400 bps：0000 4800 bps：0001 9600 bps：0010 19200 bps：0011 38400 bps：0100 57600 bps：0101 115200 bps：0110 230400 bps：0111 460800 bps：1000 912600 bps：1001	
b_9~b_{31}		Reserved	

R5030 setup example：

Baud Rate"9600"/ Stop Bit "2 bit"/ Parity Bit "None"/ Data Length "8"/ Activation "1"
$b_8b_7b_6b_5$ = 0010 / b_4 = 1 / b_3b_2 = 00 / b_1 = 1 / b_0 = 1
R5030=0010 1 00 1 $1_{(2)}$
After converted from binary to decimal, R5030=$83_{(10)}$

Ladder Diagram Example:

c. R5039 Error Code Explanation

ID	R5039	Explanation
0x00	0	Communication Success
0x01	1	Function Code error
0x02	2	Address error
0x03	3	Data Value error
0x04	4	Slave Device error
0x05	5	Answer Confirmed (ACK) When the salve device judges the command will take a long time to process, it'll return ACK to avoid TimeOut
0x06	6	Slave Device Busy
0x08	8	Memory Parity error
0x0A	10	Gateway Path Unavailable
0x0B	11	Gateway target Device failed to Respond
0xF5	245	Non-supporting function code
0xF6	246	Modbus Packet Format error

0xF7	247	COM Port assigned by R5040 is incorrect
0xF8	248	Device Busy, all communication commands paused
0xF9	249	CRC error
0xFA	250	TimeOut error
0xFB	251	Read COM Port error
0xFC	252	Write COM Port error
0xFD	253	Open COM Port error
0xFE	254	Unknown error

Modbus Writing Command

1. Figure

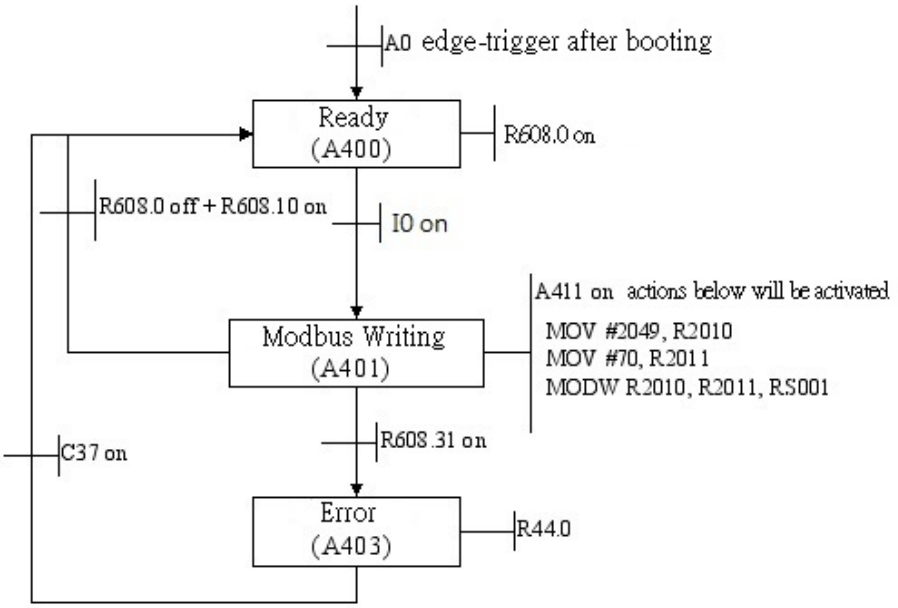
Write in single data to specific address of assigned device

2. Description
   a. Available station ID for the assigned device：0~255。
   b. MODW is MODBUS RTU16 writing component, can only write in 1 data.
   c. MODBUS RTU uses the 8-bit format, corresponding to the number of bits of the COM Port setup (Pr3942或Pr3962).
   d. The system will show the transmission state of Modbus with R608.0/.10/.31 (ref: Appendix 4-a of Modbus Reading Command), the error will be shown at R5039 if occurred during the transmission (ref: Appendix 4-c of Modbus Reading Command).
   e. Modbus COM Port is the COM Port used to set up the physical wiring, if the hardware is SUPER/10/20/200, the COM Port would be 3 (R5040=3); if it's eHMC/3/6/11/21, the COM Port would be 2 (R5040=2).
   f. Enable Modbus TCP if set R5040 to be 9, it'll be able to transmit the data through Internet (LAN), the regulations are mostly the same as Modbus RS485, below are the differences:
      i. The controller as Client (Master) and the device as Server (Slave) must be in the same domain.
      ii. The station ID of Server (Slave) should be the same as the 4th code of IP. For example, if the IP is 192.168.1.5, the station ID should set to be 5.
      iii. Able to connect with at most 2 devices as Server (Slave).
      iv. The communication timeout can be set with Pr3971.
   g. If need to change the communication protocol immediately, please apply R5030 (ref: Appendix 4-b). If R5030 is not assigned or R5030.0 not activated, please set up the standard communication protocol parameter according to the COM Port number.
      For example: If using COM2, Pr3941~Pr3950 should be set up; if using COM3, Pr3961~Pr3970 should be set up.
   h. The component contents need to be set up:
      Data Written Address：Set up the data address of external devices (Slave) to be written
      For example：Set the data written address as 2010 means the data address of external device should be written in R2010. Assume the data address is $0801_{(16)}$, then make R2010=2049 (usually the address of external devices will be shown in hexadecimal form, please convert to decimal form by yourself, $0801_{(16)}=2049_{(10)}$).
      Data Content be written：The data content written to external devices (Slave)
      For example：If the data content is set as 2011, the data content written to the external device should be filled in R2011. If the data content is 70, then make R2021=70.
      Device Address：The station ID of external devices (Slave)
      For example：If set the device address to be 1, it means the station ID of the external device is 1 (the station ID of external devices can be set by its own parameters).

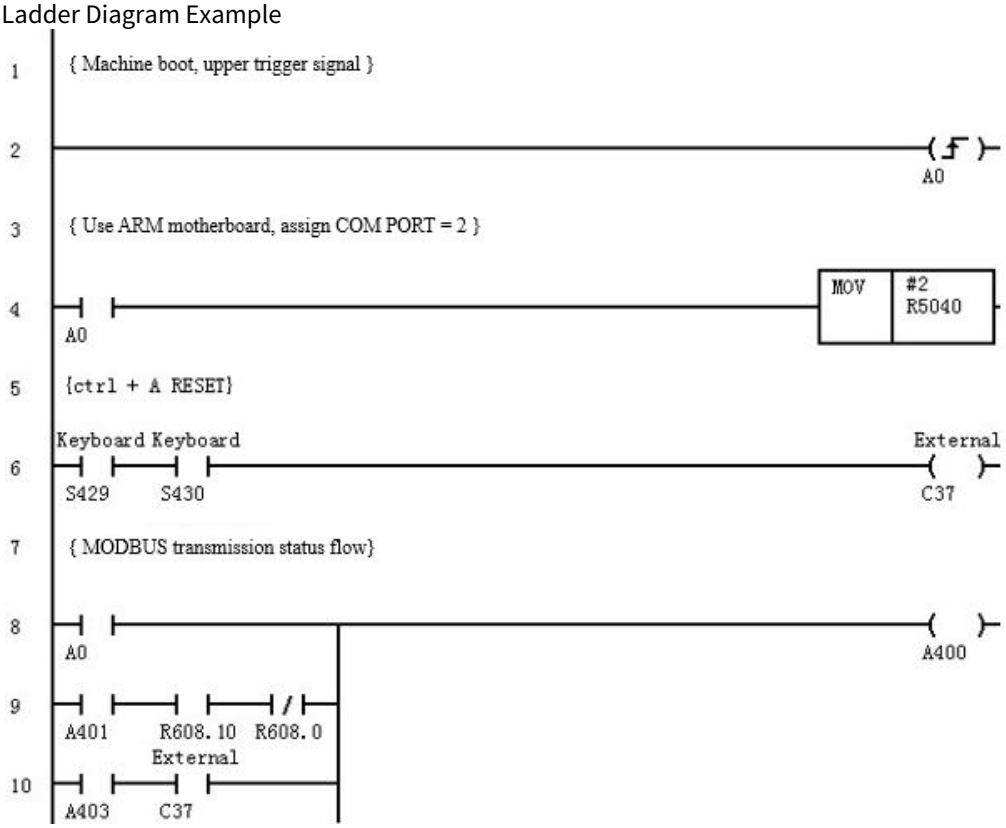
i. All the address are filled in with decimal form in the ladder diagram.
j. Please refer to the Appendix of Modbus Reading Command for all the other details

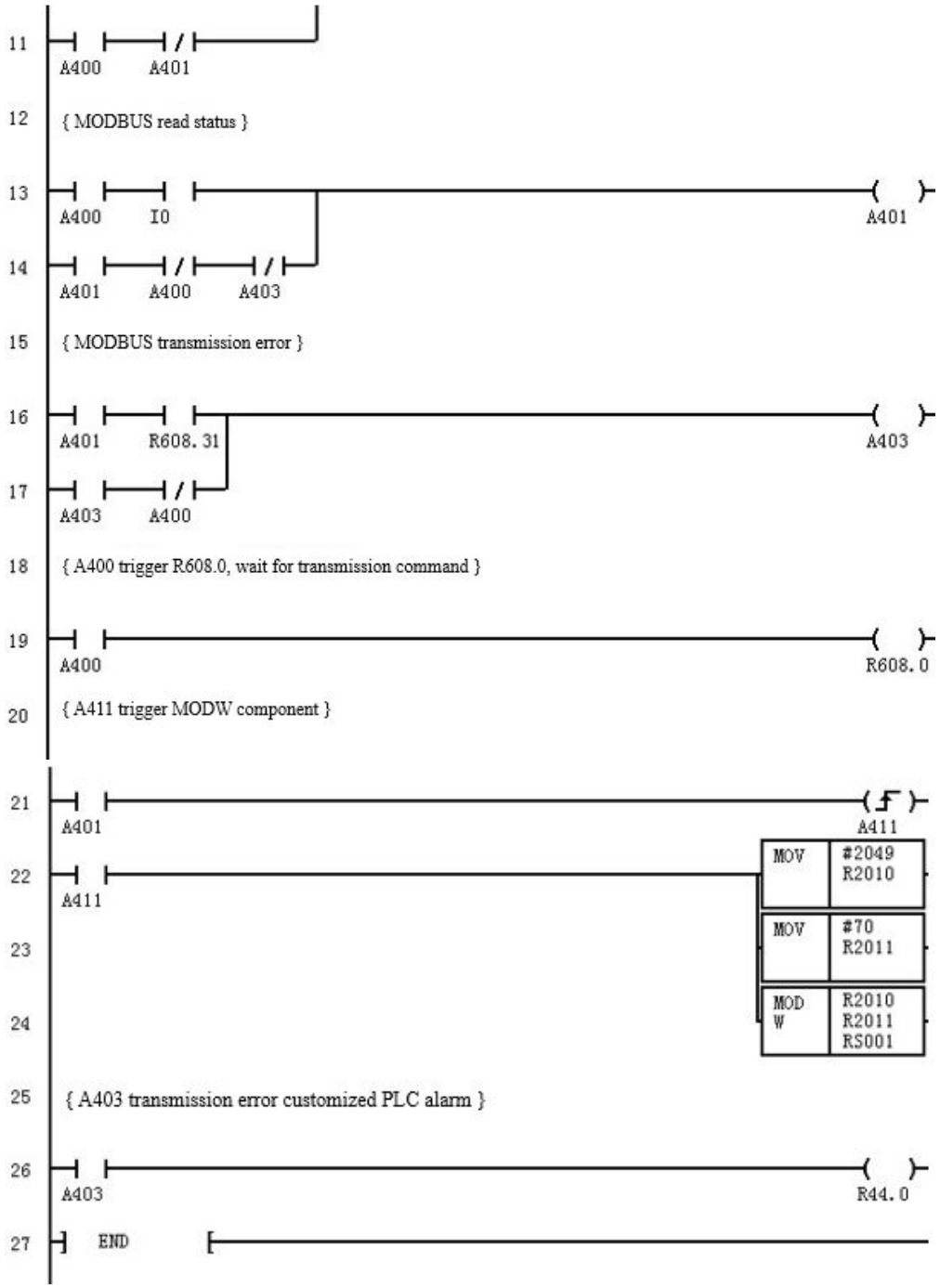
3. Example
   a. Required Specifications
      Applying eHMC controller (using COM2) for Modbus data writing communication (station ID=001, data address=0801H) with address 0801 of external device. The data written address of MODW component is R2010 (R2010=2049); the data content be written is R2011 (R2011=70). Write the stored value into the address 0801 of external device when I0 is activated, if the writing action goes properly, then it's able to see the correct value on the external address; if transmission error occurred, the transmission communication will be paused immediately and alarms will be sent, it'll return to communication ready state after pressing the Reset button.
   b. MSD
      i. A400 Communication Ready State：
         - Enter A400 state after rising edge-triggered after booting, Reset, or after the previous data transmission is confirmed to be completed.
         - A400 should keep activating R608.0
         - The data transmission completion can be confirmed with R608, when R608.0 Off and R608.10 On means the previous data is successfully transmitted (ref: Appendix 4-a of Modbus Reading Command).
      ii. A401 Modbus Writing State：
         - In A400 state, it'll enter A401 state when I0 is activated.
         - A401 will activate A411 with rising edge-trigger, trigger the MODW component once to run the Modbus writing action.
      iii. A403 Transmission Error State：
         - In A401 state, if unable to complete the Modbus action successfully, the controller will activate R608.31 automatically, which means there's a transmission handshake error. Then enters A403 transmission error state and trigger alarm R44.0 to announce the transmission error (the error code will be shown at R5039, ref: Appendix 4-c of Modbus Reading Command).
         - If triggers Reset in A403 state, it'll return to A400 state.

c. MSD



d. Ladder Diagram Example





e. IOCSAR

No.	Description	Note
I0	The I point triggering writing actions	

No.	Description	Note
C37	Reset Button	Kernel Definition
S429	Keyboard Button: Ctrl	
S430	Keyboard Button: A	
A0	Rising edge-triggered after booting	
A400	Communication Ready State	
A401	Modbus Writing State	
A403	Transmission Error State	
A411	Writing action triggering flag	
R44.0	Modbus transmission error alarm	
R608.0	Data transmission start	Kernel Definition
R608.10	Transmission completed with correct data	Kernel Definition
R608.31	Transmission Error	Kernel Definition
R2010	Data address to be written of external device	
R2011	Data content writing to the external device	
R5040	Assigned COM Port for Modbus communication	

Self–defined Modbus Command

1. Figure

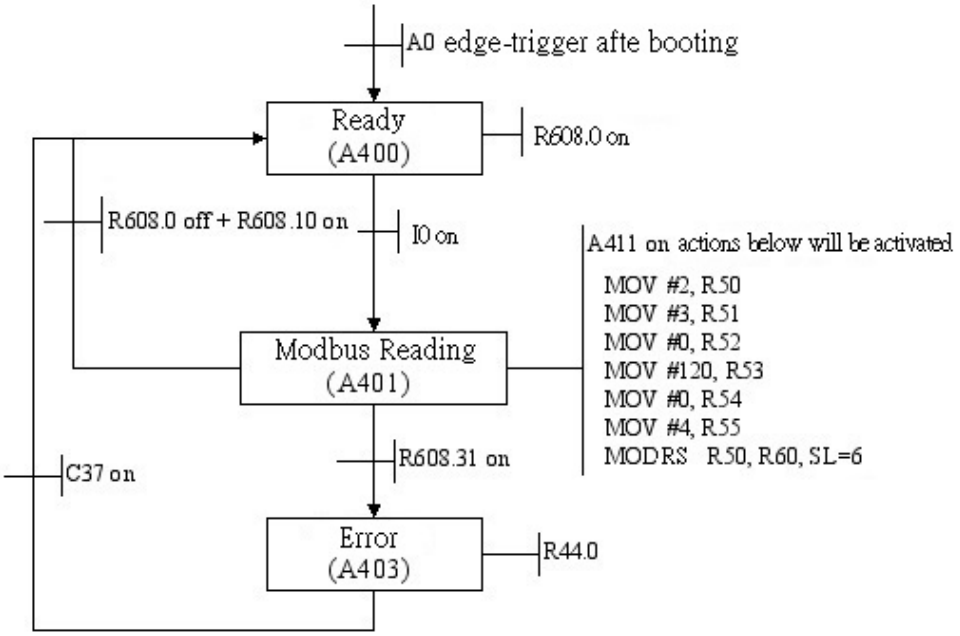
Send assigned commands by Modbus

2. Description
   a. Self-defined communication packet, only needs to write the communication content of Modbus, CRC will be done by the component.
   b. The communication format is RTU form(8-bit format), corresponding to the number of bits of the COM Port setup (Pr3942 or Pr3962).
   c. It's able to use the ModRS command for unlimited times, but can't run 2 or more ModRS commands at the same time.
   d. The ModRS command can't modify the transmitting data content during execution.
   e. The system will show the transmission state of Modbus with R608.0/.10/.31 (ref: Appendix 4-a of Modbus Reading Command), the error will be shown at R5039 if occurred during the transmission (ref: Appendix 4-c of Modbus Reading Command).
   f. Modbus COM Port is the COM Port used to set up the physical wiring, if the hardware is SUPER/10/20/200, the COM Port would be 3 (R5040=3); if it's eHMC/3/6/11/21, the COM Port would be 2 (R5040=2).
   g. Enable Modbus TCP if set R5040 to be 9, it'll be able to transmit the data through Internet (LAN), the regulations are mostly the same as Modbus RS485, below are the differences:
      i. The controller as Client (Master) and the device as Server (Slave) must be in the same domain.
      ii. The station ID of Server (Slave) should be the same as the 4th code of IP. For example, if the IP is 192.168.1.5, the station ID should set to be 5.
      iii. Able to connect with at most 2 devices as Server (Slave).
      iv. The communication timeout can be set with Pr3971.
   h. If need to change the communication protocol immediately, please apply R5030 (ref: Appendix 4-b). If R5030 is not assigned or R5030.0 not activated, please set up the standard communication protocol parameter according to the COM Port number.
   i. The component contents need to be set up:
      Initial Source Address：The initial register address of the sending data.
      For example：Set the initial source address as 50 means the sending data starts from R50.
      Initial Returning Address：The initial register address for saving the returned data.
      For example：If set the initial returning address to be 60, it means the returned data will be saved from R60.
      Length of Source Data：The length of the sending data.
      For example：Set the length of source data as 6 means the total number of the sending data is 6. Each of the data will be filled in to the source registers in decimal form. If the initial source address is set as 50, then the sending data will be filled in to R50, R51, R52, R53, R54, R55 in order.

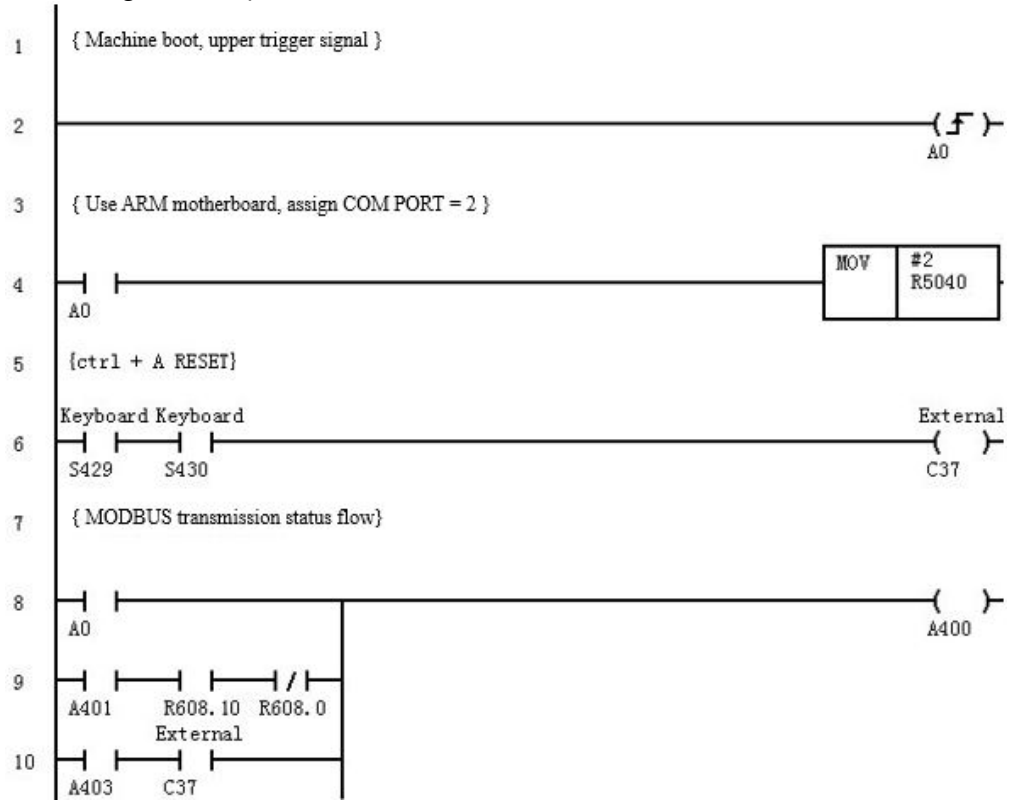
j. All the address are filled in with decimal form in the ladder diagram.
k. Please refer to the Appendix of Modbus Reading Command for all the other details.

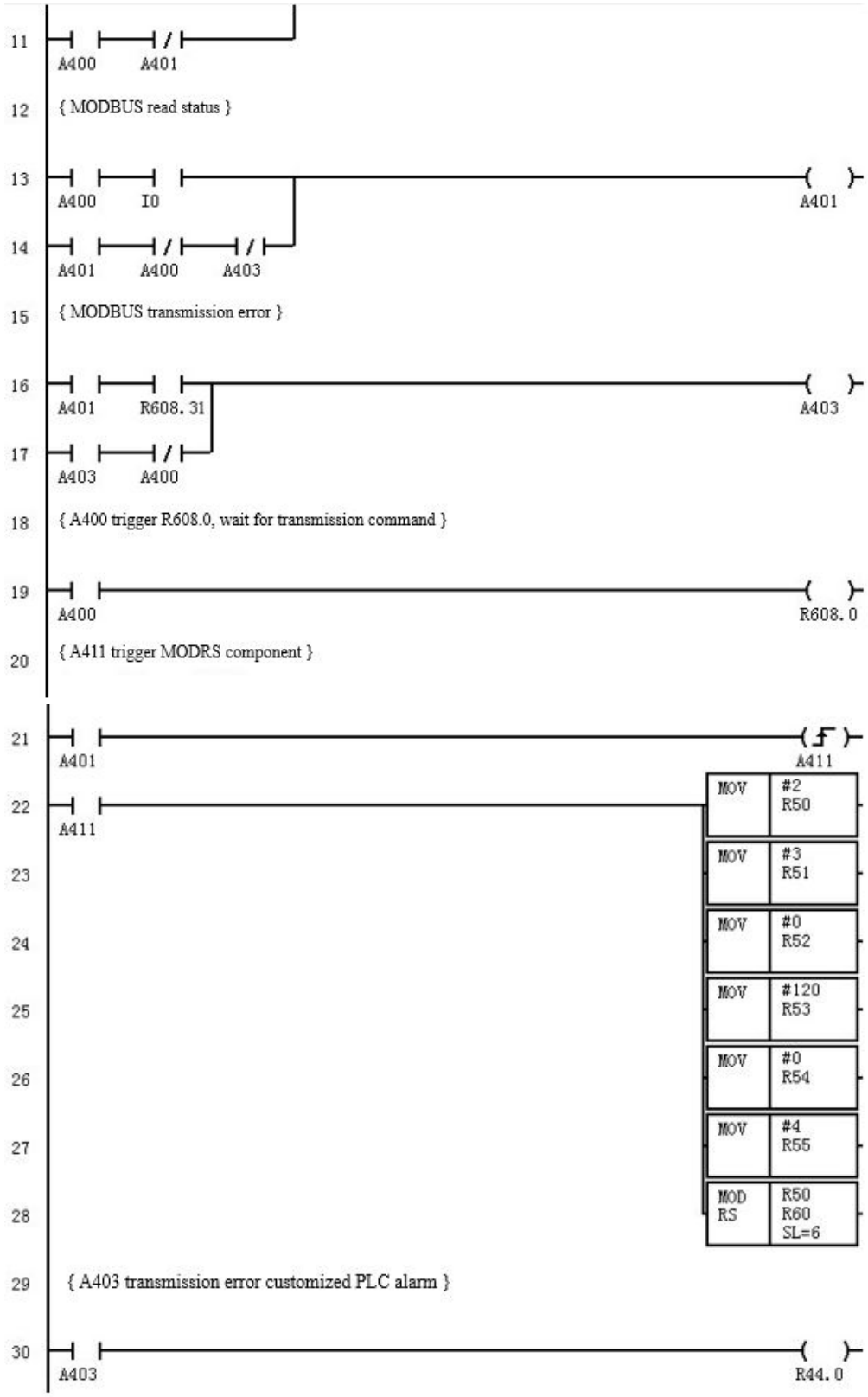
3. Example 1
   a. Required Specifications
      Applying eHMC controller (using COM2/ as Master) for Modbus data reading communication with R60 and R61(station ID=002, data address=120) of the other eHMC controller (using COM2/ as Slave).
      When I0 is activated, the command stored in Master end will be sent out. If Slave end reads the signal properly, it'll return the values of assigned registers to Master end correctly and the returned value will be saved in several registers starting from R60 (MODRS component setup in Master end：initial source address=50, which means the sending data starts from R50; initial returning address=60, which means the received data will be saved from R60; length of source data=6, which means the total number of sending data is 6).
      If transmission error occurred, the transmission will be paused immediately and alarms will be sent, it'll return to communication ready state after pressing the Reset button.
   b. Controller Setup in Slave End
      i. Parameter Setup
         Pr3234 (Activate Modbus Slave COM Port) =2
         Pr3235 (Station ID of Slave) =2
         Needs to be 8-bit format corresponding to the number of bits of the COM Port setup (Pr3942 or Pr3962).
      ii. After reboot, make sure the Modbus Slave is enabled correctly. If working normally, R5029 will equal to 0. (Modbus Slave error code R5029, ref: 5-a)
   c. Source Data Content in Master End
      According to the required specifications, refer to Appendix 5-b. multiple R value reading function, assign R50=2, R51=3, R52=0, R53=120, R54=0, R55=4
   d. MSD
      i. A400 Communication Ready State：
         - Enter A400 state after rising edge-triggered after booting, Reset, or after the previous data transmission is confirmed to be completed.
         - A400 should keep activating R608.0
         - The data transmission completion can be confirmed with R608, when R608.0 Off and R608.10 On means the previous data is successfully transmitted (ref: Appendix 4-a of Modbus Reading Command).
      ii. A401 Modbus Reading State：
         - In A400 state, it'll enter A401 state when I0 is activated.
         - A401 will activate A411 with rising edge-trigger, trigger the MODRS component once to run the Modbus reading action.
      iii. A403 Transmission Error State：
         - In A401 Modbus reading state, if unable to complete the Modbus action successfully, the controller will activate R608.31 automatically, which means there's a transmission handshake error. Then enters A403 transmission error state and triggers alarm R44.0 to announce the transmission error (the error code will be shown at R5039, ref: Appendix 4-c of Modbus Reading Command).
         - If triggers Reset in A403 state, it'll return to A400 state.

e. MSD



f. Ladder Diagram Example




11
A400
A401
12
{ MODBUS read status }
13
A400
I0
A401
14
A401
A400
A403
15
{ MODBUS transmission error }
16
A401
R608. 31
A403
17
A403
A400
18
{ A400 trigger R608.0, wait for transmission command }
19
A400
R608. 0
20
{ A411 trigger MODRS component }
21
A401
A411
22
A411
23
24
25
26
27
28
MOV #2 R50
MOV #3 R51
MOV #0 R52
MOV #120 R53
MOV #0 R54
MOV #4 R55
MOD RS R50 R60 SL=6
29
{ A403 transmission error customized PLC alarm }
30
A403
R44. 0

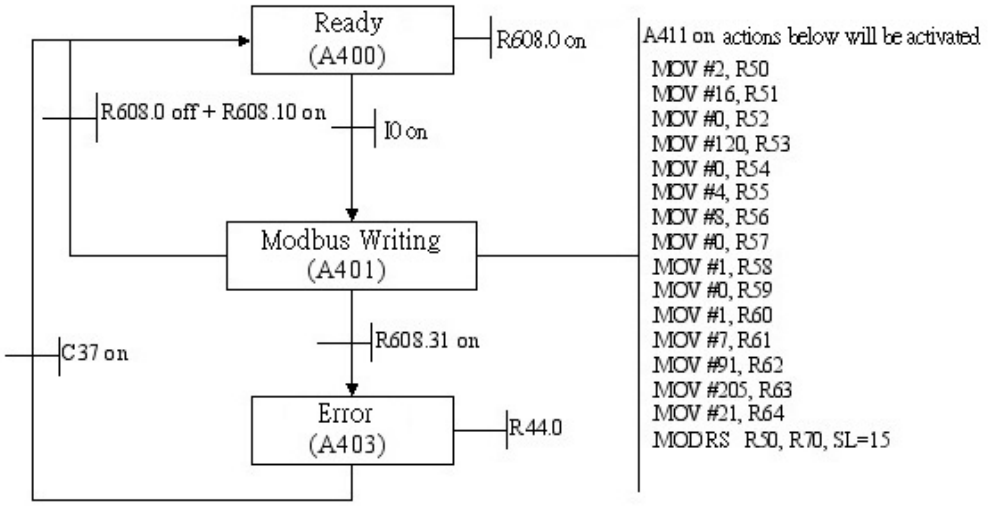
g. IOCSAR

No.	Description	Note
I0	The I point triggering Modbus actions	
C37	Reset Button	Kernel Definition
S429	Keyboard Button: Ctrl	
S430	Keyboard Button: A	
A0	Rising edge-triggered after booting	
A400	Communication Ready State	
A401	Modbus Reading State	
A403	Transmission Error State	
A411	Reading action triggering flag	
R44.0	Modbus transmission error alarm	
R608.0	Data transmission start	Kernel Definition
R608.10	Transmission completed with correct data	Kernel Definition
R608.31	Transmission Error	Kernel Definition
R50	Initial register address of the sending data	
R60	Initial register address for saving the returned data.	
R5040	Assigned COM Port for Modbus communication	

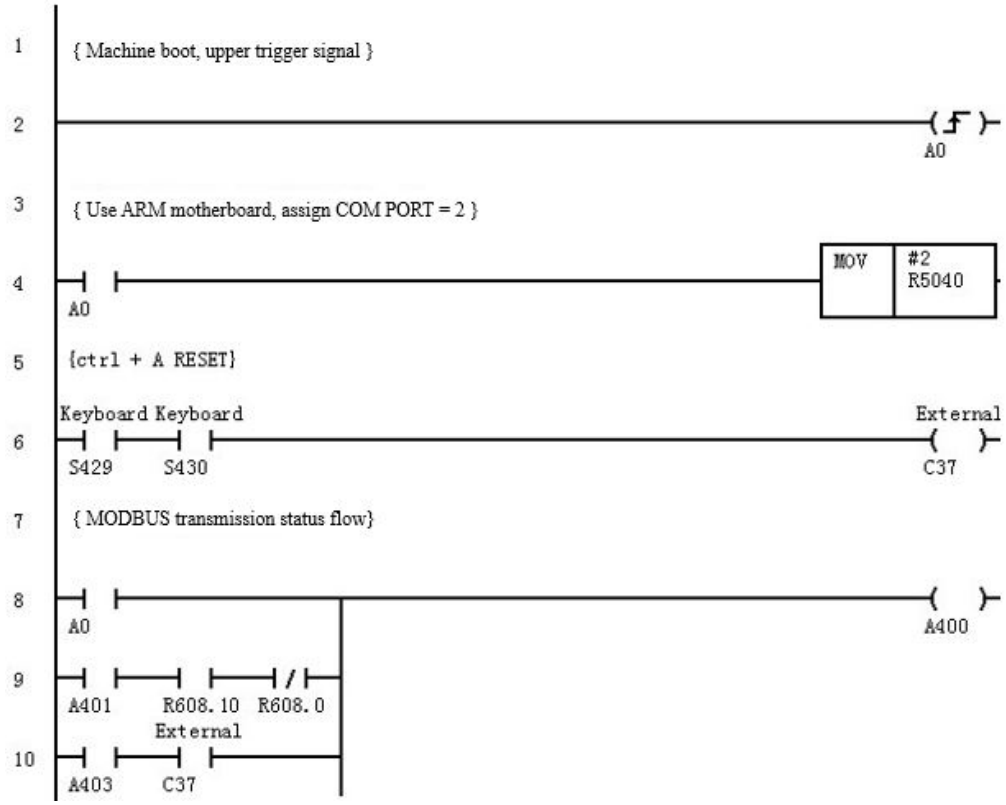
4. Example 2

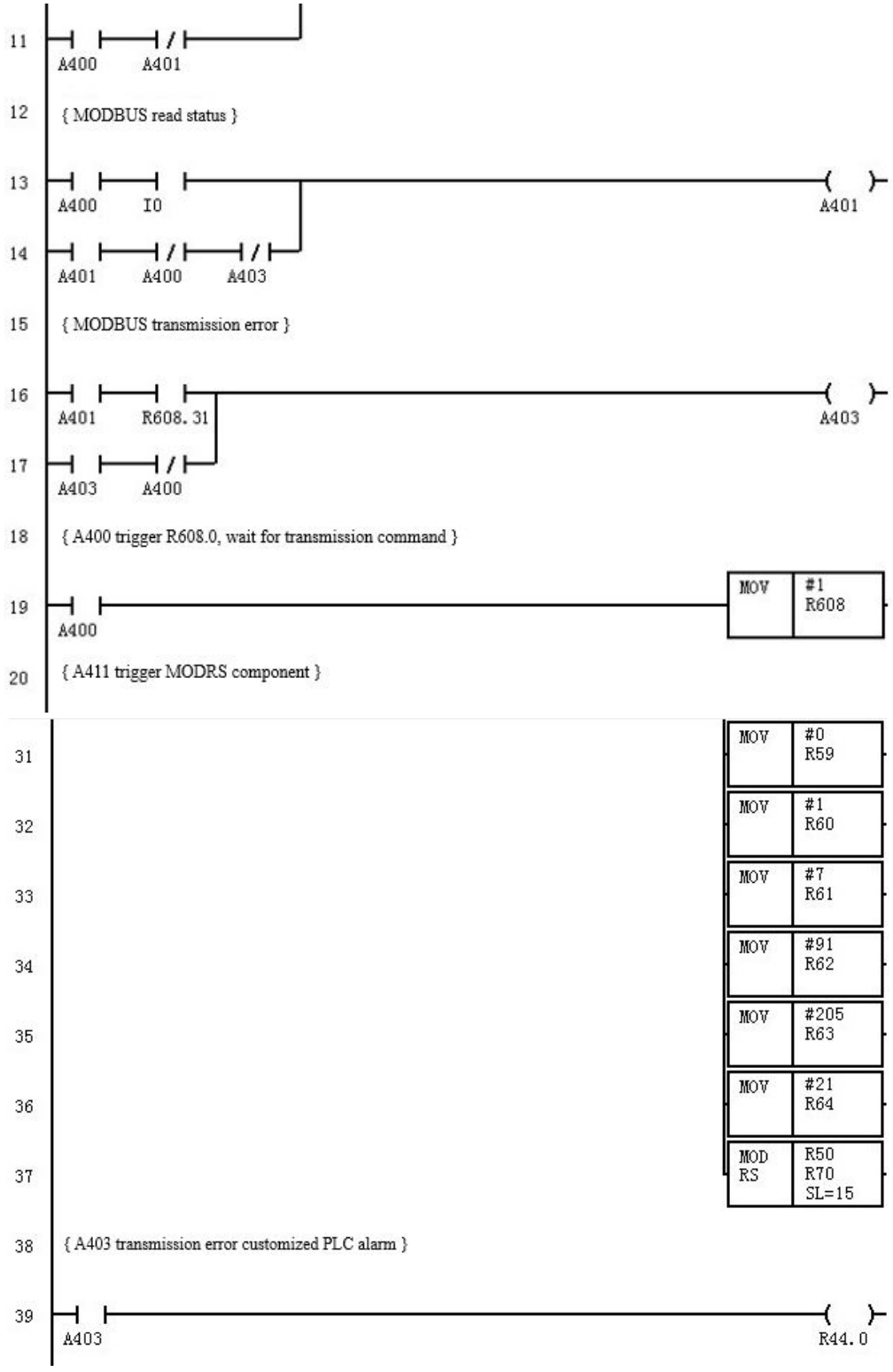
a. Required Specifications
   Applying eHMC controller (using COM2/ as Master) for Modbus data writing communication with R60 and R61(station ID=002, data address=120) of the other eHMC controller (using COM2/ as Slave).
   When I0 is activated, the command stored in Master end will be sent out. If the Slave end receives the signal properly, values will be written into R60 and R61 then return the correct return value. The return value will be saved in several registers of Master end starting from R70 (MODRS component setup in Master end：initial source address=50, which means the sending data starts from R50; initial returning address=70, which means the received value will be saved from R70; length of source data=15, which means the total number of sending data is 15).
   If transmission error occurred, the transmission will be paused immediately and alarms will be sent, it'll return to communication ready state after pressing the Reset button.
b. Controller Setup in Slave End
   i. Parameter Setup
      Pr3234 (Activate Modbus Slave COM Port) =2
      Pr3235 (Station ID of Slave) =2
      Needs to be 8-bit format corresponding to the number of bits of the COM Port setup (Pr3942 or Pr3962).
   ii. After reboot, make sure the Modbus Slave is enabled correctly. If working normally, R5029 will equal to 0. (Modbus Slave error code R5029, ref: 5-a)
c. Source Data Content in Master End
   According to the required specifications, refer to Appendix 5-c. multiple R value writing function, assign R50=2, R51=16, R52=0, R53=120, R54=0, R55=4, R56=8, R57=0, R58=1, R59=0, R60=1, R61=7, R62=91, R63=205, R64=21
d. MSD
   i. A400 Communication Ready State：
      - Enter A400 state after rising edge-triggered after booting, Reset, or after the previous data transmission is confirmed to be completed.
      - A400 should keep activating R608.0
      - The data transmission completion can be confirmed with R608, when R608.0 Off and R608.10 On means the previous data is successfully transmitted (ref: Appendix 4-a of Modbus Reading Command).
   ii. A401 Modbus Writing State：
      - In A400 state, it'll enter A401 state when I0 is activated.
      - A401 will activate A411 with rising edge-trigger, trigger the MODRS component once to run the Modbus writing action.
   iii. A403 Transmission Error State：
      - In A401 Modbus writing state, if unable to complete the Modbus action successfully, the controller will activate R608.31 automatically, which means there's a transmission handshake error. Then enters A403 transmission error state and triggers alarm R44.0 to announce the transmission error (the error code will be shown at R5039, ref: Appendix 4-c of Modbus Reading Command).
      - If triggers Reset in A403 state, it'll return to A400 state.

e. MSD



f. Ladder Diagram Example




11
A400
A401
12
{ MODBUS read status }
13
A400
I0
A401
14
A401
A400
A403
15
{ MODBUS transmission error }
16
A401
R608. 31
A403
17
A403
A400
18
{ A400 trigger R608.0, wait for transmission command }
19
A400
MOV
#1
R608
20
{ A411 trigger MODRS component }
31
MOV
#0
R59
32
MOV
#1
R60
33
MOV
#7
R61
34
MOV
#91
R62
35
MOV
#205
R63
36
MOV
#21
R64
37
MOD
RS
R50
R70
SL=15
38
{ A403 transmission error customized PLC alarm }
39
A403
R44. 0

g. IOCSAR

No.	Description	Note
I0	The I point triggering Modbus actions	
C37	Reset Button	Kernel Definition
S429	Keyboard Button: Ctrl	
S430	Keyboard Button: A	
A0	Rising edge-triggered after booting	
A400	Communication Ready State	
A401	Modbus Writing State	
A403	Transmission Error State	
A411	Writing action triggering flag	
R44.0	Modbus transmission error alarm	
R608.0	Data transmission start	Kernel Definition
R608.10	Transmission completed with correct data	Kernel Definition
R608.31	Transmission Error	Kernel Definition
R50	Initial register address of the sending data	
R70	Initial register address for saving the returned data.	
R5040	Assigned COM Port for Modbus communication	

5. Appendix
    a. Registers about Modbus Slave communication(R5029)

Error Code	R5029	Explanation
0x00	0	Slave Activation Success
0x01	1	COM Port Open Failure
0x02	2	Illegal COM Port Number (1~3)
0x03	3	Illegal Station ID (0~255)
0x04	4	Incorrect Packet Length
0x05	5	Incorrect Packet Content
0x06	6	CRC error of packet delivered by Master
0x07	7	NC file upload suspension command error not received
0x08	8	NC File Reading Error
0x09	9	Return Failure

b. Multiple Register Reading Function

i. Transmit：

- Data transmitting format and example
  Station ID (1) + Function Code (1) + Initial reading address (2) + Number of Reading Data (2)
  P.S. The number in the ( ) represents the number of data or the occupying Bytes of the data.

  Example：The station ID of Slave is 2, reading 2 registers from Slave, R60 & R61, the initial source address of Master end starts from R50, R50~R55 total 6 data.

Station ID (1)	**Function Code (1)**	**Initial Reading Address (2)**	**Number of Reading Data (2)**
2	3	0、120	0、4
R50=2	R51=3	R52=0 R53=120	R54=0 R55=4

- Data Explanation

Data Explanation	Number of Data (Occupying Bytes)	Command Value / Range	Explanation
Station ID	1 Byte	0~255	
Function Code	1 Byte	3	MODBUS standard function code 0x03
Initial Reading Address*	2 Bytes	Register：R0~R30000 Corresponding initial reading address：0~60000 (decimal)	Initial reading address = Register number*2 Example： The initial reading address of register is R60, so the initial reading address =60*2=120 Fill in 0, 120 following to the format
Number of Reading Data (N)	2 Bytes	1~125	It needs 2 reading data for a register of controller. Example： Needs 4 reading data when reading R60, R61 Fill in 0, 4 following to the format

Calculation of Initial Reading Address*

(1) Calculate the initial reading address first.

(2) Judge if it's bigger than 255, if it's smaller or equals to 255, set the first byte 0 and fill in the second byte with the initial address.

(3) If it's bigger than 255, convert to hexadecimal form first then separate into former and latter values, 2 digits for each.

(4) Convert the former and latter values into decimal form respectively then fill in for use.

Example 1： R60

a. Initial Reading Address =60 x 2=$120_{(10)}$
b. 120 is smaller than 255, the first and second byte will be 0, 120 respectively.

Example 2： R6000

a. Initial Reading Address =6000 x 2=$12000_{(10)}$

b. 12000 is bigger than 255, requires conversion
c. $12000_{(10)}$=$2EE0_{(16)}$, separate into $2E_{(16)}$, $E0_{(16)}$
d. Convert 2E and E0 into decimal form respectively, so the first and second byte will be 46, 224.

ii. Return：

- Date returning format and example
  Station ID (1) + Function Code (1) + Number of bytes (1) + Data Vale (2 x N)
  N= number of reading data
  P.S. The number in the ( ) represents the number of data or the occupying bytes of the data.

  Example：The station ID of Slave is 2, reading 2 registers from Slave, R60=65537 and R61=123456789. The return values are stored in the returning address from R60 of Master end, R60~R70, total 11 data.

Station Id (1)	**Function Code (1)**	**Number of Bytes (1)**	**Data Value (2xN)**
2	3	8	0、1、0、1、7、91、205、21
R60=2	R61=3	R62=8	R63=0 R64=1 R65=0 R66=1 R67=7 R68=91 R69=205 R70=21

- Data Explanation

Data Explanation	Number of Data (Occupying Bytes)	Command Value / Range	Explanation
Station ID	1 Byte	0~255	
Function Code	1 Byte	3	

Number of Bytes	1 Byte	2N	N=the number of reading data Example： If the number of reading data is set as 4 before transmitting, the number of bytes will be 4 x 2=8
Data Value*	2N Bytes	0~255	Judges the number of data according to the number of bytes, every data will be converted to decimal form after read by the controller.

Calculation of data value*

(1) A register of the controller will be separated into 4 data values for transmission, the value get first is high-byte and the latter is low-byte.

(2) For example, for values a, b, c, d, they should be calculated in the order from left to right: $a*16^6+b*16^4+c*16^2+d*16^0$

Example 1：R value =65537, the data values will be 0, 1, 0, 1

a. The return values are 0, 1, 0, 1 in order.
b. Calculate with the formula: $0*16^6+1*16^4+0*16^2+1*16^0=65537$

Example 2：R value =123456789, the data values will be 7, 91, 205, 21

a. The return values are 7, 91, 205, 21 in order.
b. Calculate with the formula: $7*16^6+91*16^4+205*16^2+21*16^0=123456789$

c. Multiple Register Writing Function

i. Transmitting：

- Data transmitting format and example

  Station ID (1) + Function Code (1) + Initial Writing Address (2) + Number of Writing Data (2) + Number of Bytes of Writing Data (1) + Writing Data value (2N)

  N=number of writing data

  P.S. The number in the ( ) represents the number of data or the occupying bytes of the data.

  Example：The station ID of Slave is 2, writing to 2 registers of Slave, the writing values are R60=65537 and R61=123456789 respectively. The initial source address of Master end starts from R50, R50~R64 total 15 data.

Station ID (1)	**Function Code (1)**	**Initial Writing Address (2)**	**Number of Writing Data (2)**
2	16	0、120	0、4

R50=2	R51=16	R52=0 R53=120	R54=0 R55=4

Number of Bytes of Writing Data (1)	**Writing Data Value (2N)**
8	0、1、0、1、7、91、205、21
R56=8	R57=0、R58=1 R59=0、R60=1 R61=7、R62=91 R63=205、R64=21

- Data Explanation

Data Information	Number of Data (Occupying Bytes)	Command Value / Range	Explanation
Station ID	1 Byte	0~255	
Function Code	1 Byte	16	
Initial Writing Address*	2 Bytes	Register：R0~R30000 Corresponds to the initial writing address：0~60000 (decimal)	Initial writing address = register number*2 Example： The initial writing address of the register is R60, corresponding to the initial writing address =60*2=120 Fill in 0, 120 following to the format.
Number of Writing Data	2 Bytes	2~122	It needs 2 writing data to writing to a register of controller. Example： It needs 4 writing data to write value into R60, R61. Fill in 0,4 following to the format.

Number of Bytes of Writing Data	1 Byte	2N	N=Number of writing data Example： If the number of writing data is set as 4, the number of bytes will be 4 x 2=8
Data Value**	2N Bytes	0~255	It needs 4 data for a register of controller Example： It needs 8 data to write R60, R61.

Calculation of Initial Writing Address*

(1) Calculate the initial writing address first.

(2) Judge if it's bigger than 255 ,if it's smaller or equals to 255, set the first byte as 0 and fill in the second byte with initial writing address.

(3) If it's bigger than 255, convert to hexadecimal form first then separate into former and latter values, 2 digits for each.

(4) Convert the former and latter values into decimal form respectively then fill in for use.

Example 1： R60

1. Initial Writing Address =60 x 2=$120_{(10)}$
2. 120 is smaller than 255, the first and second byte will be 0, 120 respectively.

Example 2： R6000

1. Initial Writing Address =6000 x 2=$12000_{(10)}$
2. 12000 is bigger than 255, requires conversion
3. $12000_{(10)}$=$2EE0_{(16)}$, separate into $2E_{(16)}$, $E0_{(16)}$
4. Convert 2E and E0 into decimal form respectively, so the first and second byte will be 46, 224.

Calculation of Data Value**

(1) A register of the controller will be separated into 4 data values for transmission, the value written in first is high-byte and the latter is low-byte.

(2) For example, with the written value x and being separated into 4 data values a, b, c, d

$x/16^6$ =a……a1

$a1/16^4$ =b……b1

$b1/16^2$ =c……c1

c1=d

a is the highest order, d is the lowest order, put a, b, c, d in order as written value.

Example 1：For writing in the value of R=65537, the data values will be 0, 1, 0, 1.

1. Calculate with the formula

$65537/16^6$ =0……65537（a=0）

$65537/16^4$ =1……1（b=1）

$1/16^2$ =0……1（c=0）

1=d

a=0、b=1、c=0、d=1

Example 2：For writing in the value of R=123456789, the data values will be 7, 91, 205, 21

1. Calculate with the formula

$123456789/16^6$ =7……6016277（a=7）

$10668473/16^4$ =91……52501（b=91）

$52501/16^2$ =205……21（c=205）

21=d

a=7、b=91、c=205、d=21

ii. Returning：

- Data returning format and example
  Station ID (1) + Function Code (1) + Initial Writing Address (2) + Number of Writing Data (2)
  P.S. The number in the ( ) represents the number of data or the occupying bytes of the data.

  Example：The station ID of Slave is 2, writing to 2 registers of Slave, the writing value are R60=65537 and R61=123456789. The returned data are stored in the returning address from R70 of Master end, R70~R75 total 6 data.

Station ID (1)	**Function Code (1)**	**Initial Writing Address (2)**	**Number of Writing Data (2)**
2	16	0、120	0、4
R70=2	R71=16	R72=0 R73=120	R74=0 R75=4

- Data Explanation

Data Explanation	Number of Data	Command Value / Range	Explanation

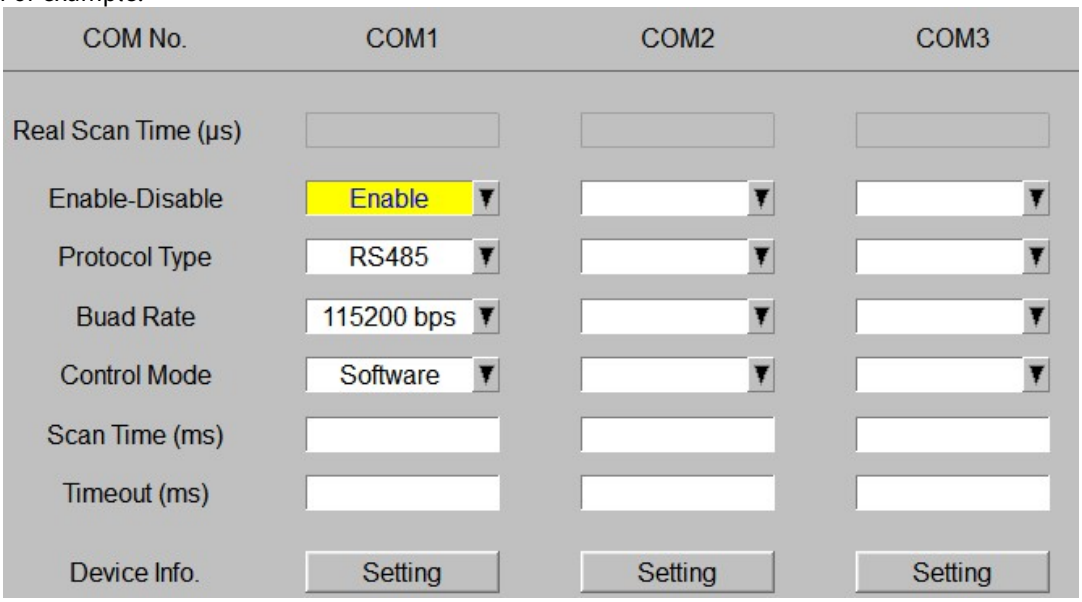
	(Occupying Bytes)		
Station ID	1 Byte	0~255	The returned values are the same as the transmitted values, used for confirmation.
Function Code	1 Byte	16	
Initial Writing Address	2 Bytes	Register： R0~R30000 Corresponds to the initial writing address：0~60000 (decimal)	
Number of Writing Data	2 Bytes	2~122	

Self-defined Modbus Command of SRI

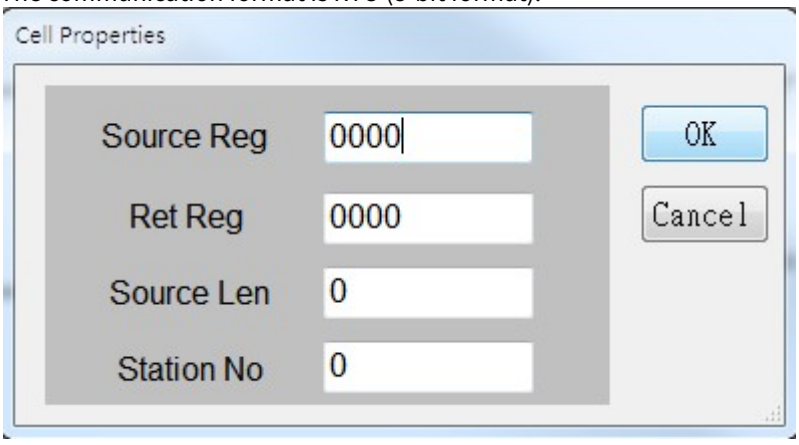
1. Figure

Send self-defined Modbus commands through SRI port

2. Description
   a. The Control Mode of SRI should be set as **Software Mode,** so the function can operate properly. For example:

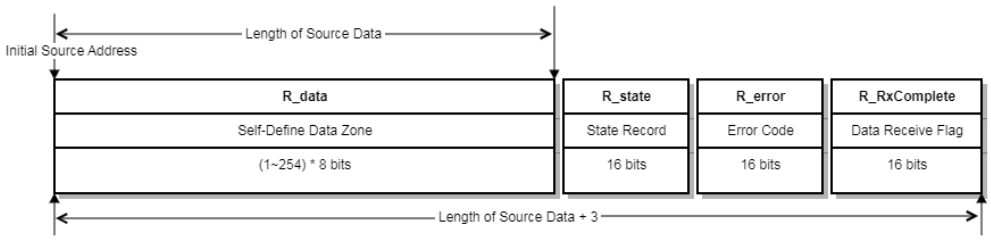


   b. SRI port of the controller only supports the controller as Master, please refer to Modbus series componens introduction if it's required to be Slave.
   c. Self-defined packet of SRI only needs to setup the communication content of Modbus when applying, CRC will be done by the component.
   d. The communication format is RTU (8-bit format).



e. Initial Source Address(Source Reg)：Setup the initial register of the transmitting packet, 8 bits for each register. Please note that the last 3 registers must be reserved, $\mathbf{R_{state}}$, $\mathbf{R_{error}}$ and $\mathbf{R_{RxComplete}}$

f.



g. Initial Returning Address(Ret Reg)：Setup the initial register of the returning packet, 8 bits for each register. Please reserves the required registers according to the returning packet.
h. Length of Source Data(Source Len)：According to the packet format of Modbus and deduct the length after CRC, therefore it's 254 bytes at maximum.
i. Device Address(Station No)：Corresponds to the COM Port of SRI.
j. It's able to use SRI command for unlimited times in the ladder program, but can't run 2 or more SRI commands at the same time.
k. SRI command can't modify the content of transmitting data during execution.
l. The system will show the communication state at $\mathbf{R_{state}}$, details are shown below：

Value	Function Explanation
0 (0x00)	Waiting to transmit
1 (0x01)	Transmitting
2 (0x02)	Waiting to receive
3 (0x03)	Receiving
16 (0x10)	Communication Failure, the error code is the value of $\mathbf{R_{error}}$.

- When communication failed, the second last register "$\mathbf{R_{error}}$" of the assigned data source address will record the corresponding error code. (ref: R5039)
- After receiving the return value, the last register "$\mathbf{R_{RxComplete}}$" will be 1, it'll stay 0 at the rest of the time.
- If the command is sent successfully, the state recorded by $\mathbf{R_{error}}$ and $\mathbf{R_{RxComplete}}$ will be initialized.

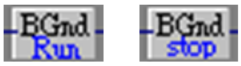
3. Example
   a. Please refer to the part about Master in ModRS example.

b. The packet format, setup methods are the same, the differences are listed below (transmission state, error code, completion flag)：

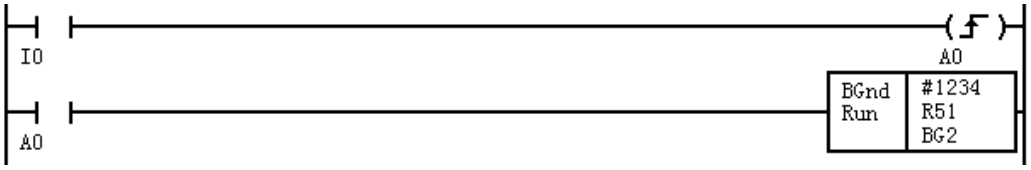
		Modbus Series Component	SRI Component
Start the data transmission		**R608.0 on**	Starts the transmission when SRI component activated and $\mathbf{R_{state}=0}$
Transmission State	Waiting to transmit		$\mathbf{R_{state}=0}$
	Transmitting	**R608.1 on**	$\mathbf{R_{state}=1}$
	Waiting to receive	**R608.2 on**	$\mathbf{R_{state}=2}$
	Receiving	**R608.3 on**	$\mathbf{R_{state}=3}$
	Transmission Error	**R608.31 on**	$\mathbf{R_{state}=16}$
Completion Flag (transmission completed with correct data)		**R608.10 on**	$\mathbf{R_{RxComplete}=1}$
Error Code		**R5039**	$\mathbf{R_{error}}$

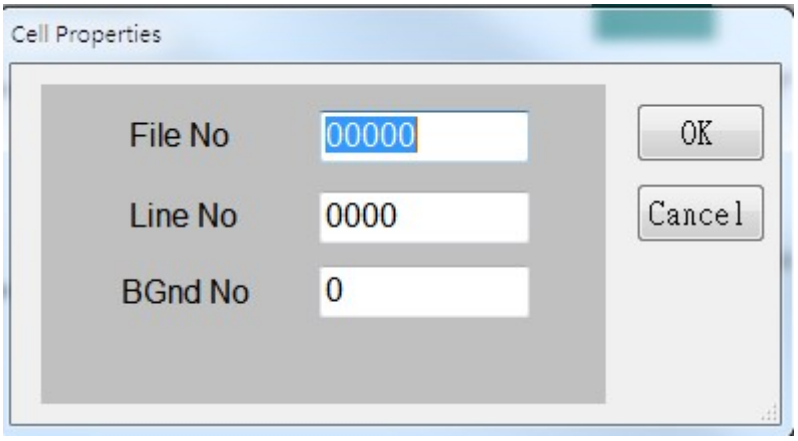
2.2.12 Background Running Component

1. Figure



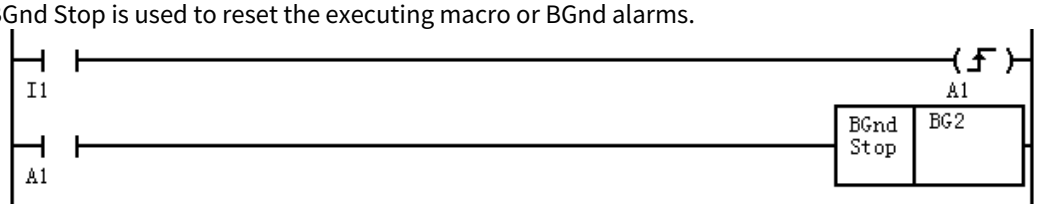
2. Description
   a. BGnd Run is used to execute the assigned macro program in ladder diagram. When the component is activated, the assigned macro will be executed, the applied arguments are introduced below：

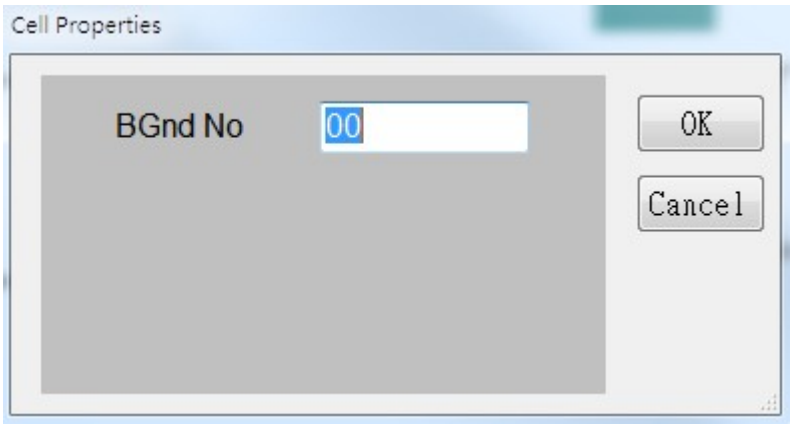




   i. Filename Number：Number of the macro (O0001~O9999)
   ii. Line Number Register (Rn)：Records the line number executing in the macro, please input the register number (R1~R65535). Rn+1 is the alarm notification.
      1. Example：Set the line number register to be 51, when alarm is triggered, R52 will change to 1 from 0 till the alarm is cleared by BGnd Stop. It's recommended to use this flag to confirm the macro execution is completed.
   iii. BGnd Number：Setup the number of BGnd component, currently supports BG00~BG19, 20 in total.

b. BGnd Stop is used to reset the executing macro or BGnd alarms.





i. BGnd Number：Input the number of component needed to Reset (BG00~BG19).

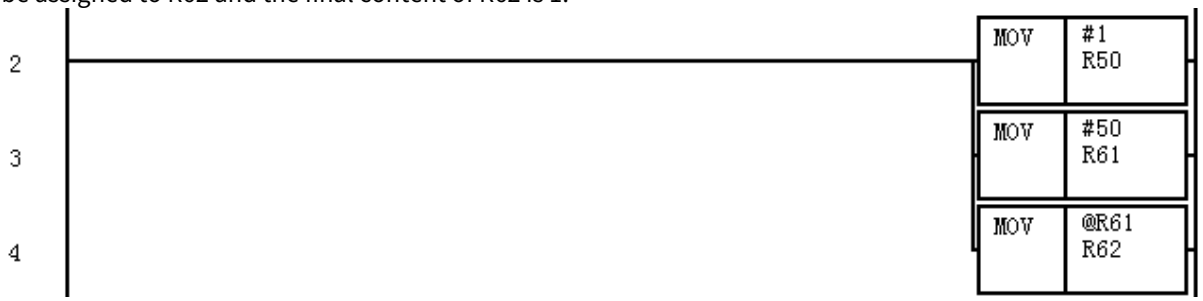
3. Limitations
   a. The version valid for PLC editors starts from 2.25.0, and needs to cooperate with controllers after version 10.116.22.
   b. Provides **20 BGnd** devices (BG00~BG19).
   c. The name of the macro must be started with **"O"**, but no need to write "O" in the ladder diagram; the name of the macro is always a 4-digits natural number (nnnn).
   d. Path of macro： DiskC\OpenCnc\Ncfiles 或 DiskC\OpenCnc\MACRO
   e. Limitation of Macro content：
      i. **Only supports the following G code**
         - **G04**
         - **G10 L1000**
         - **G10 L1801**
         - **Other G code is not supported**
      ii. **Only supports the following function**
         1. **Mathematical functions (e.g. ABS, SIN, COS, MAX...)**
         2. **SETDO**
         3. **Other function is not supported**
      iii. **Not supporting M, S, T, B code**
      iv. **Not supporting Increment Mode**
      v. **Only supports local variables #1~#400 for reading and writing.**
      vi. **Support global variable @ , the specifications are the same as regular macros.**
   f. Please refer to the alarm manual for BGnd related alarms.
      i. When alarms happened, the number of the device which caused the alarm will be shown.
      ii. When using too many BGND components, it might cause lack of memory and the alarm will happen.
   g. The component can only be in the general scanning area, the scanning speed will still be the same even it's being moved to the rapid scanning area.
   h. Please execute this component with rising edge-trigger to avoid repeated executions.
   i. After the component being activated and starts the execution, if being activated second time before completion, the second activation command will be ignored.

j. Reset and Emergency Stop won't stop BGND components, only BGnd Stop is able to stop or reset the components.
k. The BGnd Number must not be reapted, otherwise it will cause abnormal actions.

2.2.13 Indirect Addressing Function of Register

For some specific components that could applies with registers, if input the register number adding "@" can activate the indirect addressing function.

As the example shown below, the content of R61 is **50**, when scanning to the 4th line, the content of **R50** will be assigned to R62 and the final content of R62 is 1.

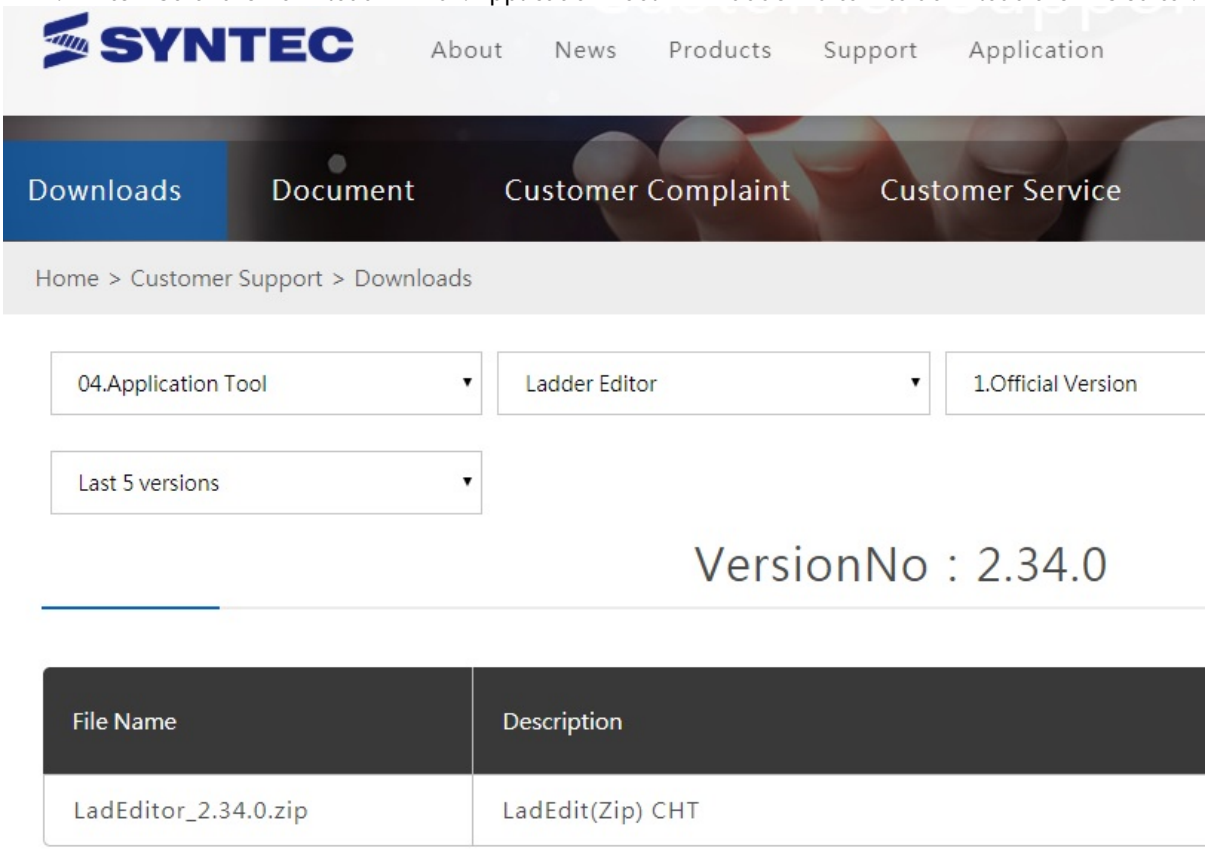


3 PLC Editor Operation Instruction.

3.1 PLC Editor Instruction

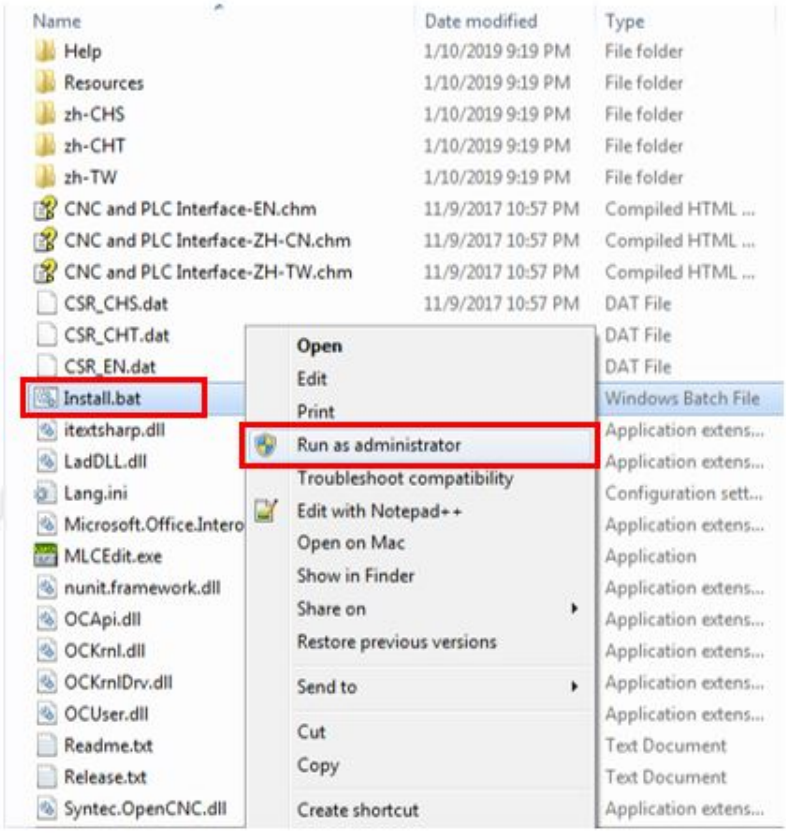
3.1.1 Software Download

1. Enter SYNTEC website http://www.syntecclub.com.tw and complete the online registration procedure, make sure the application is approved.
2. Enter "Software Download" => "04. Application Tool" => "Ladder Editor" to download the PLC editor.



3.1.2 Software Installation

- Versions before 2.26.3 （included） are typical installation version（need to be installed）
  - Run LadEdit_vx.xx.x.msi and follow the instructions to complete the installation.
  - After the installation is completed, click "Start" => "Programs" => "OpenCNC" => "MLCEdit.exe" to open the PLC editor.
- Versions after 2.29.0 （included） are xcopy installation version（no need to be installed）
  a. **When first run or after changing the file path:**
     i. After downloading LadEditor_x.xx.x.zip, right click "Install.bat" after unzipping the file, select "Run as administrator" to set PLC editor as the default program to open .lad files.
     ii. Run "MLCEdit.exe" to open the PLC editor.

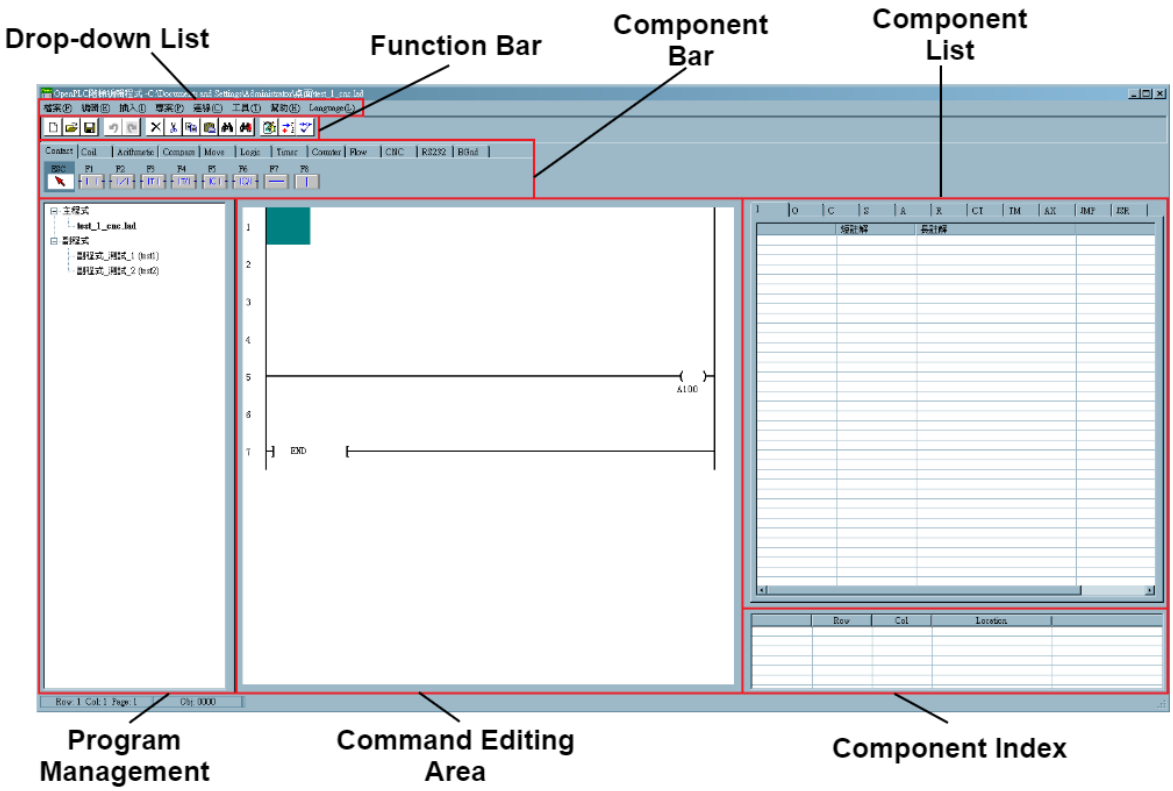


  b. **Second run and after:** Running "MLCEdit.exe" or double clicking the .lad file are both able to open the PLC editor.

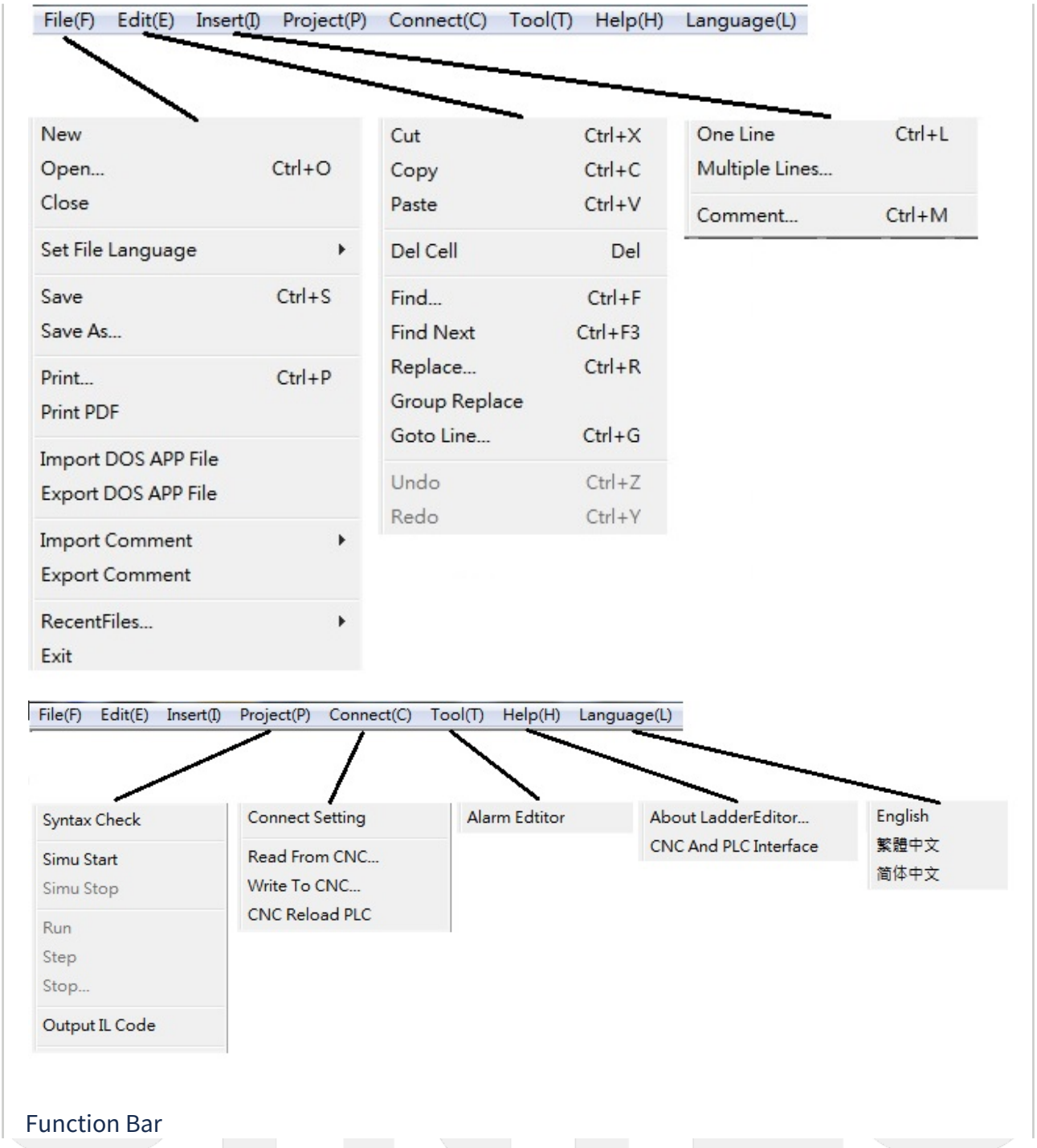
3.1.3 Interface Operation

There are total 7 sections on the PLC editor interface, shown as the picture below：

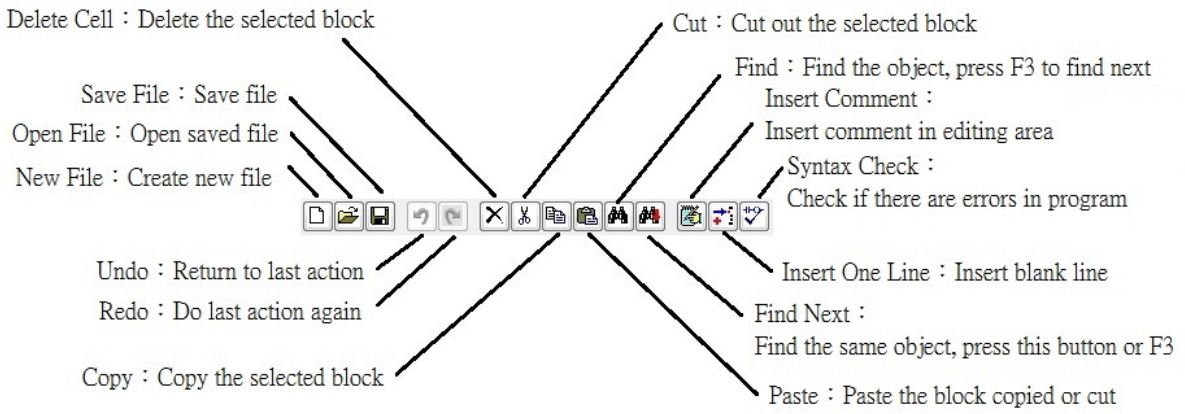
- Drop-down List
- Function Bar
- Component Bar
- Program Management
- Command Editing Area
- Component List
- Component Index



Drop-down List

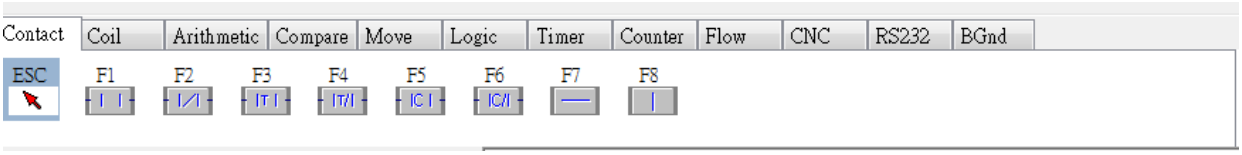


Function Bar



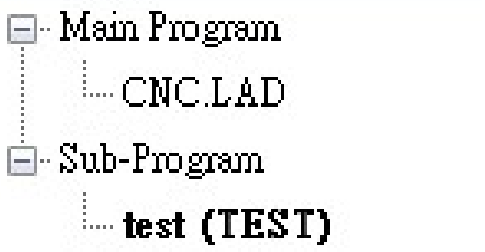
Component Bar

The section provides various kinds of components for users to select and apply, please refer to the chapters behind for further details.



Program Management

Shows the main and sub program in the current file.



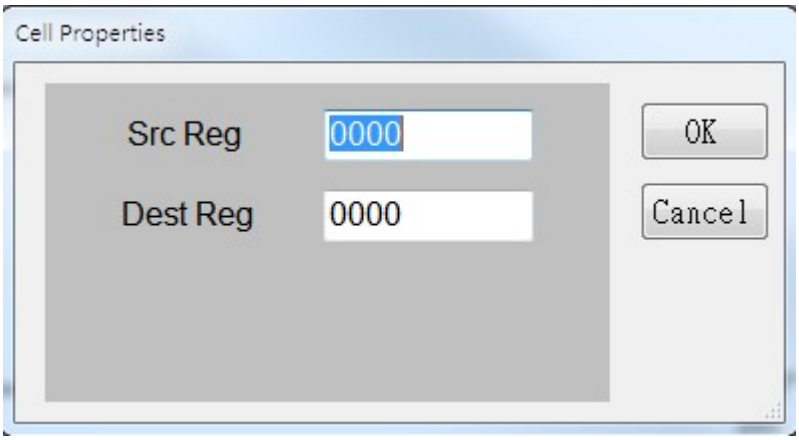
Command Editing Area

This section is used to edit the content of ladder diagram.

Parameter Input

After selecting the component from the component bar, left click on the blank space in the editing area, a parameter set-up window like the picture below will pop-up.

Click the OK button after entering the required value, the component will be written into the assigned location.



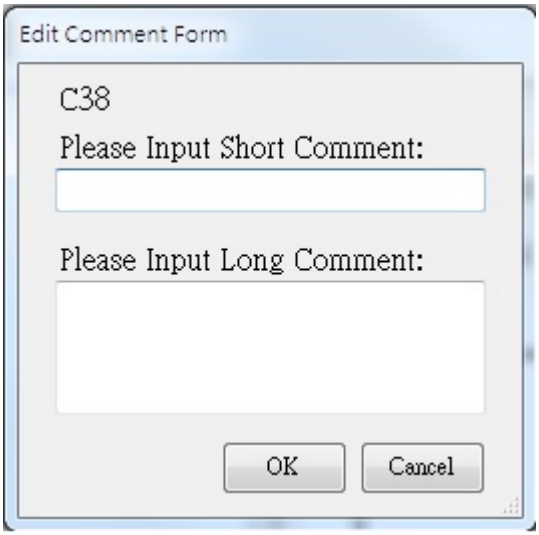
Comment Editing

After selecting the component in the editing area, a list will show up if right clicking on it.

Now choose "Edit XXX comment"（XXX is the component title）, the comment editing window will pop-up.

Length limitation of short comment is 8 characters, which could be shown above the component in editing area.

There is no limitation for long comment but it won't be shown in editing area.



Keyboard Operations

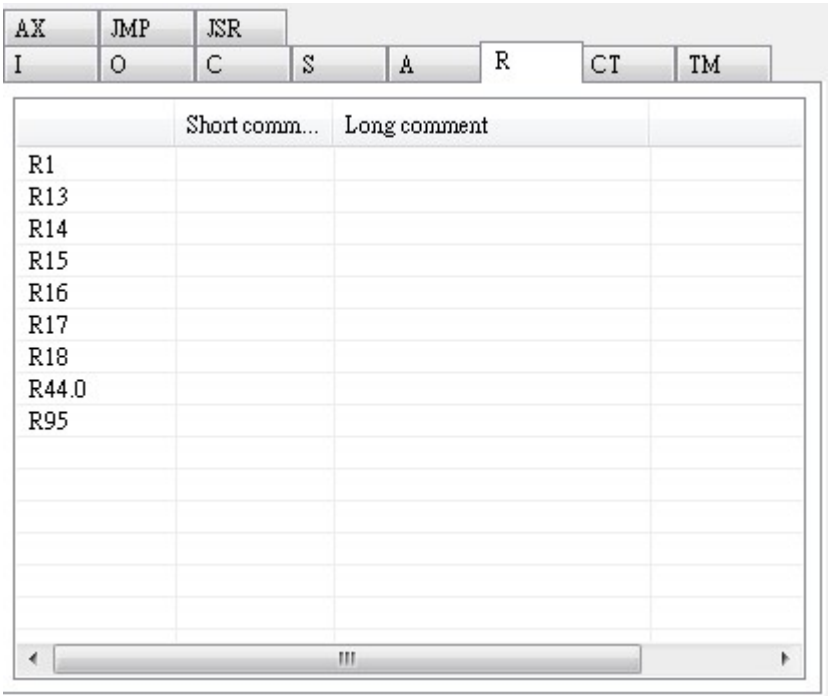
Button	Function
Page Up	Move to the previous page of cursor location while editing
Page Down	Move to the next page of cursor location while editing

Button	Function
Home	Move cursor to the left end
End	Move cursor to the right end
Delete	Delete the component at the cursor location
↑↓	Move up/down when the cursor is in editing area
← →	Move left/right when the cursor is in editing area

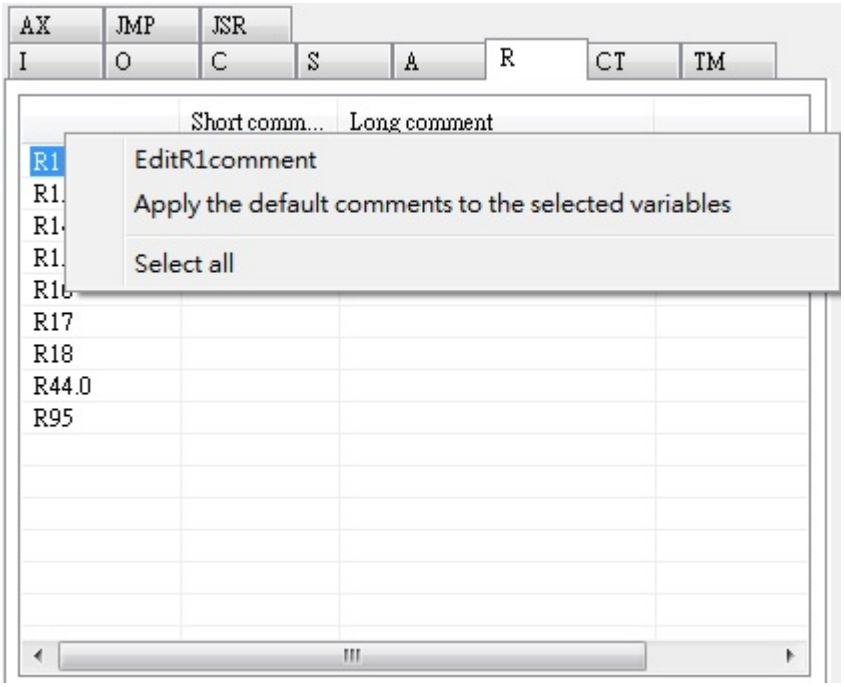
Component List

This section is the component usage list, users can check the component status here, there are several categories:

I、O、C、S、A、R、CT、TM、AX、JMP、JSR



Right click on any component on the list can also edit the comment.



Component Index

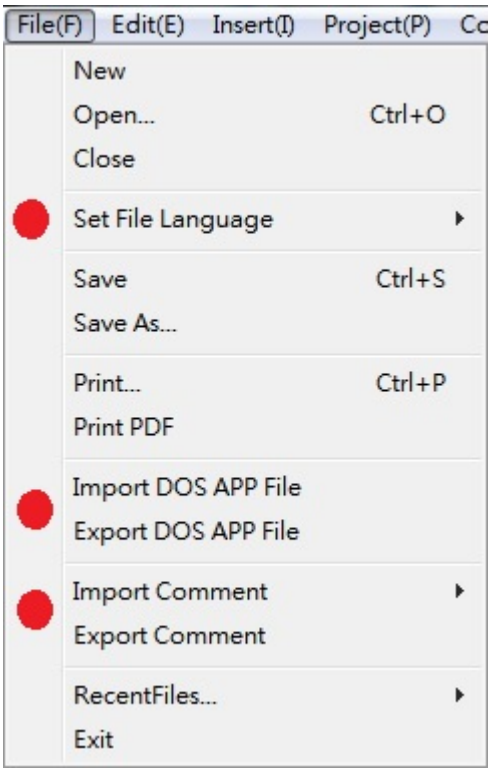
Operation Steps：

1. Click any component in the editing area, this section will list the address of the component, like coordinates. All of the components in the main/sub program are included.
2. Double left click on specific address in the list, the cursor will jump to the correspond location. If the address located in sub program, the cursor also jumps to sub program.

	Row	Col	Location
C36	57	5	CNC.LAD
C36	62	7	CNC.LAD
C36	67	7	CNC.LAD
C36	76	10	CNC.LAD
C36	76	2	CNC.LAD
C36	77	2	CNC.LAD
C36	82	1	CNC.LAD
C36	92	6	CNC.LAD

3.1.4 Function Introduction

File Operation



Set File Language

If the ladder diagram is generated by old version PLC editor, the language of interface might occur error （Traditional/ Simplified Chinese） when opening up with new version PLC editor.

This function is able to set up the language of the file.

This problem won't be found in ladder diagrams generated by PLC editors after version 2.26.3 （included）, so there is no need to use the function.

Import/Export DOS APP File

This function is able to import/export the long/short comment files of DOS system, Traditional/Simplified Chinese file name: APPCHI.STR; English filename: APPENG.STR.

Import Long/Short Comment

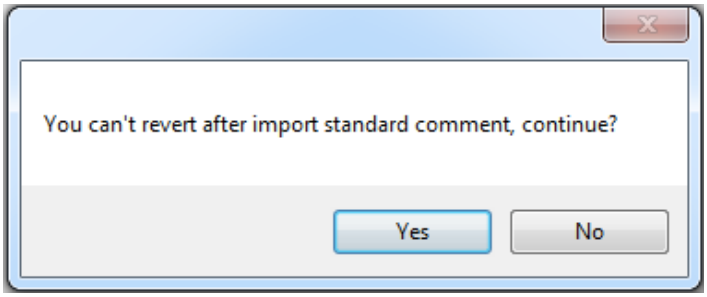
This function is used to import new version comment files （different from DOS）.

Please note, if there are customized comments in the current ladder diagram, they must be exported before importing external comment files or the content will be overwritten.

There are 2 sources of comment to import：

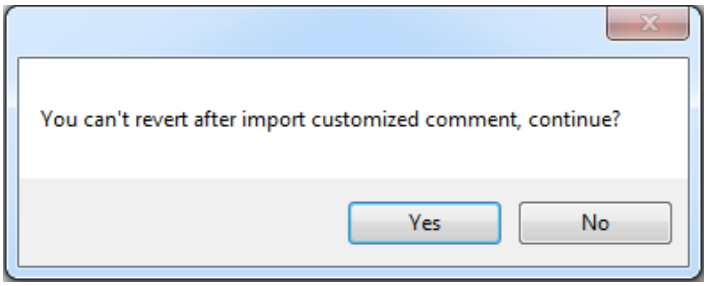
- Import standard comment

Confirmation window will pop-up, click "Yes" to import standard comments.

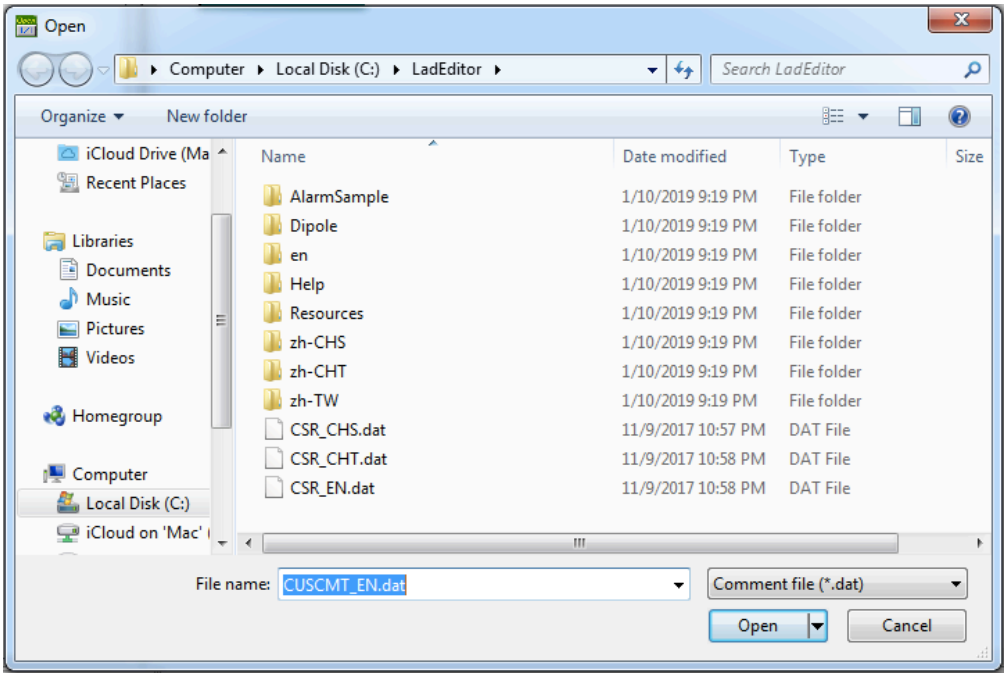


- Import customized comment

Confirmation window will pop-up, click "Yes" to import customized comments.



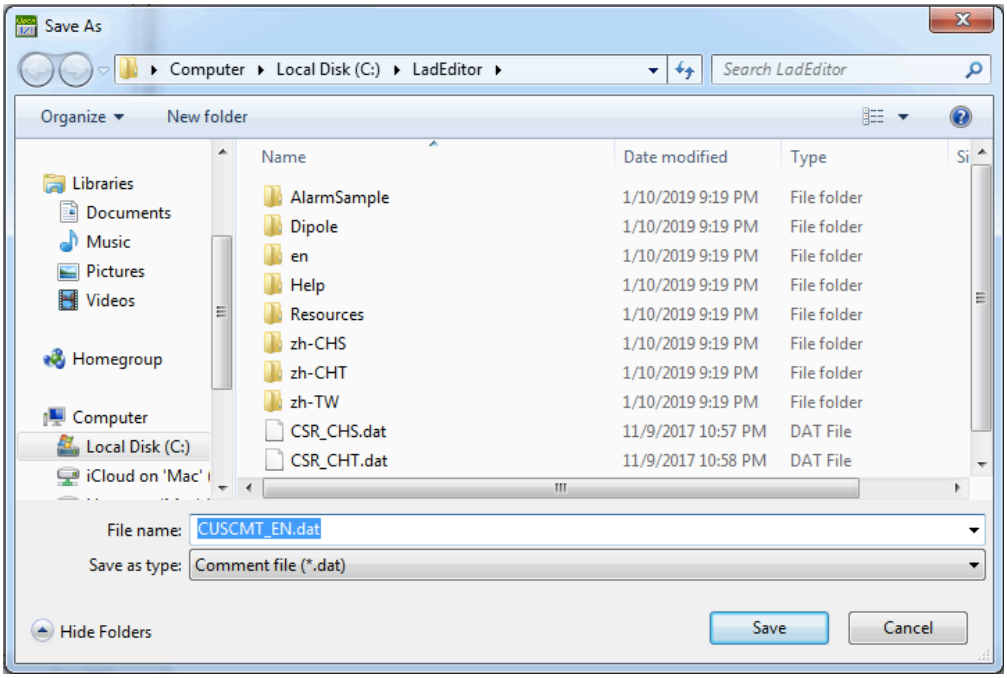
System window will pop-up after pressing "Yes", please choose the importing file.



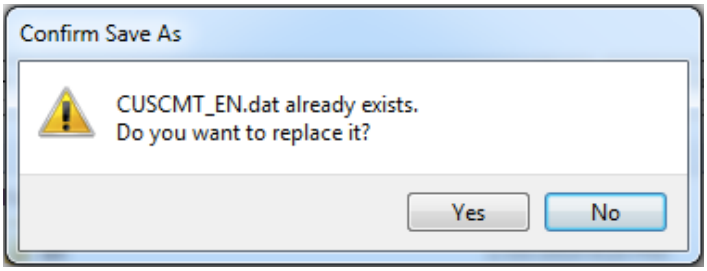
Export Long/Short Comments

This function is used to export comments, so they can be transferred to other ladder diagrams for use. The contents can be long/short comments.

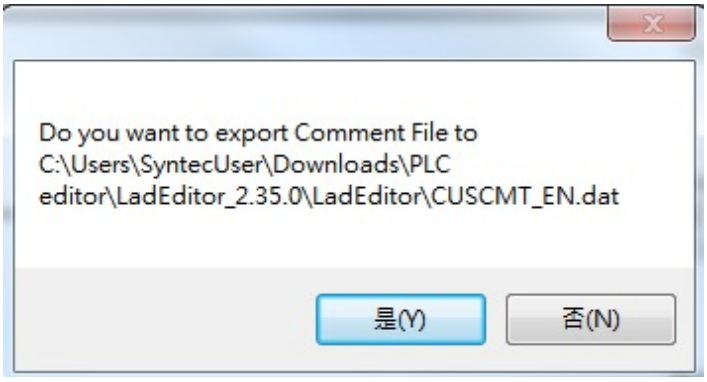
System window will pop-up when applying this function, users are able to set up the export location and file name.



System reminder will pop-up if assigned the existed file name.



The saving path of the file will be shown on the screen while exporting for users to make final confirmation.

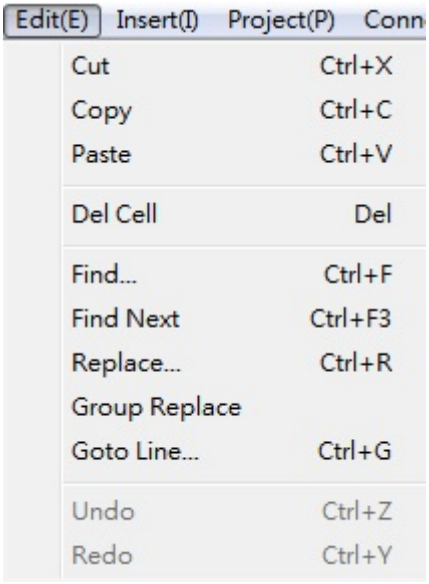


The default exporting path of the comment file is the running location of PLC editor, which is the LadEditor folder.

The default filename of customized comment is CUSCMT_**L**.dat, L is the abbreviation of language, currently supporting Traditional Chinese（CHT）, Simplified Chinese（CHS）, English（EN）.

The filename of comment exported under CHT interface is CUSCMT_CHT.dat; under CHS interface is CUSCMT_CHS.dat。

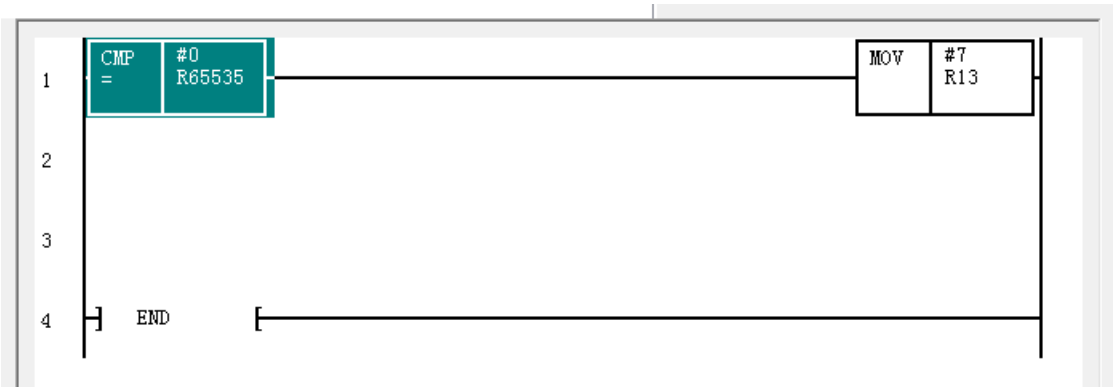
Editing Operation



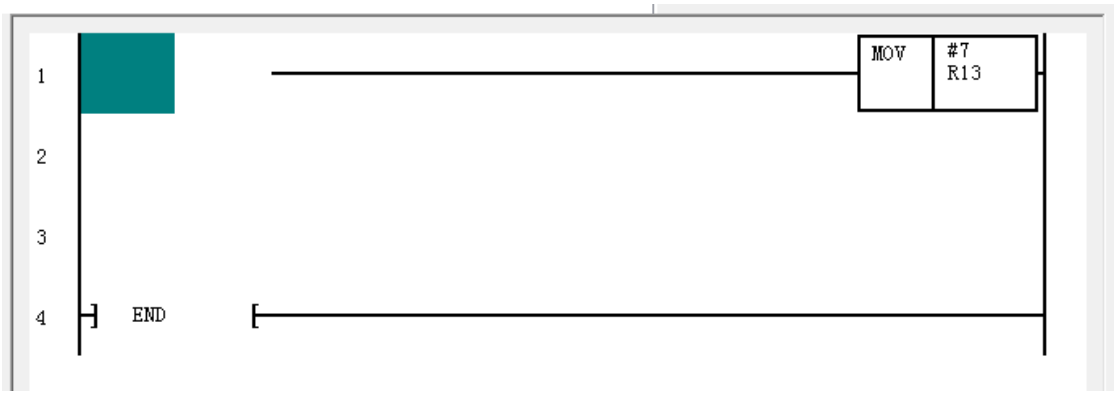
Cut / Copy / Paste / Delete Cell

Cut Cell | Copy Cell | Paste Cell | Delete Cell

- Move cursor to the cell

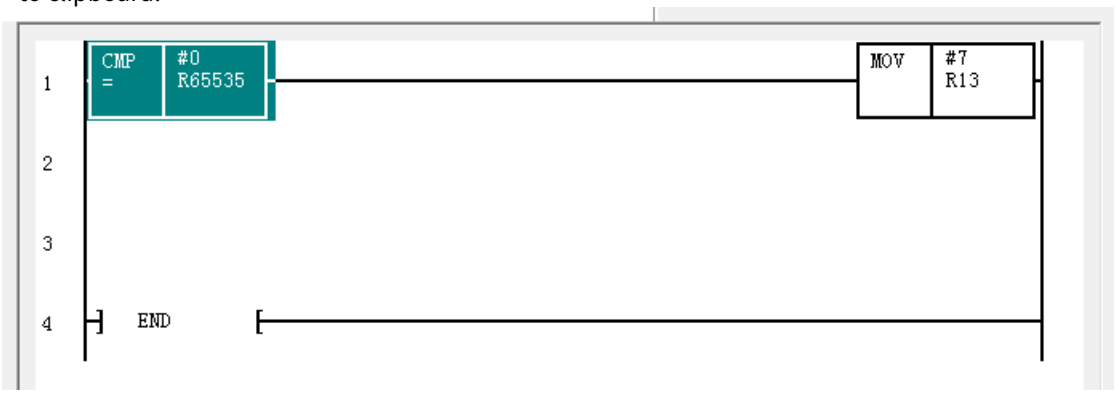


- Click "Cut" in the "Edit" above or press Ctrl+X on keyboard to cut down the cell and save to clipboard.



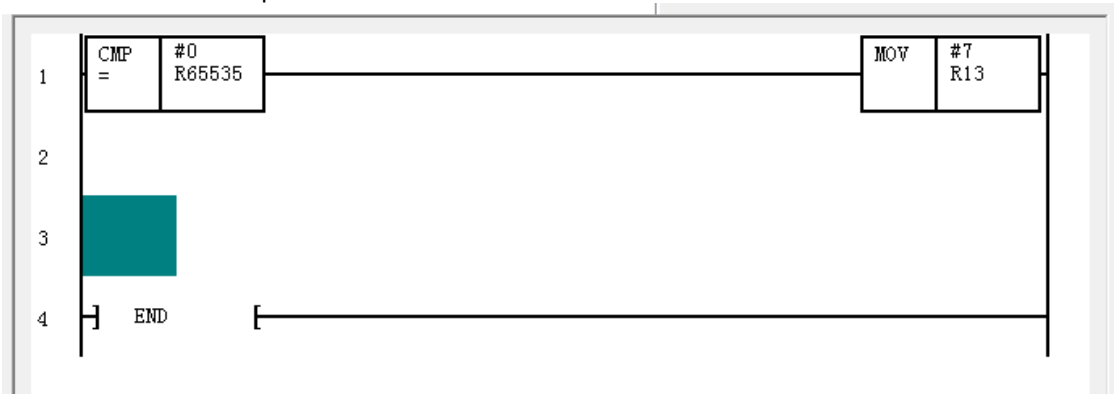
Cut Cell Copy Cell Paste Cell Delete Cell

- Move cursor to the cell, click "Copy" in the "Edit" above or press Ctrl+C on keyboard to copy the cell to clipboard.

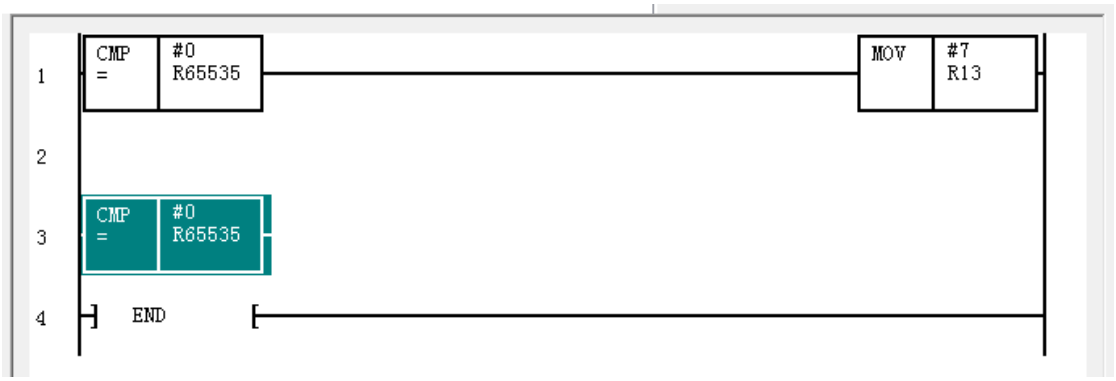


Cut Cell Copy Cell Paste Cell Delete Cell

- Move cursor to the required location.

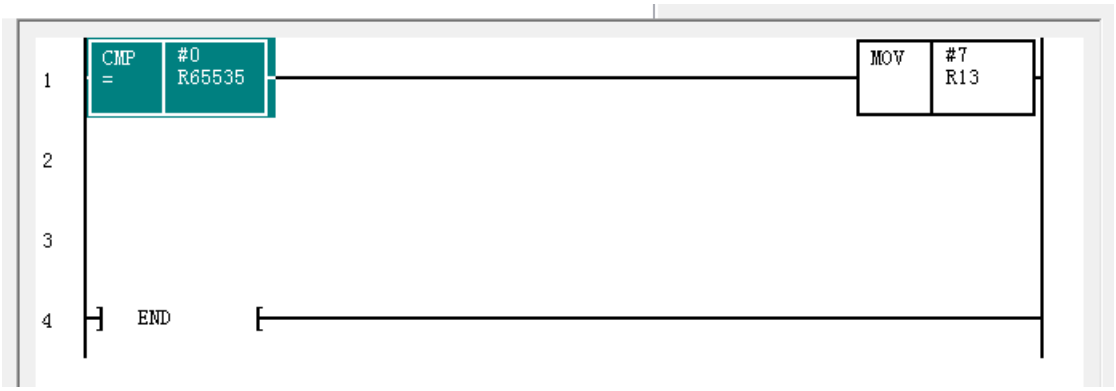


- Click "Paste" in the "Edit" above or press Ctrl+V on keyboard to paste the cell in the clipboard to the location.

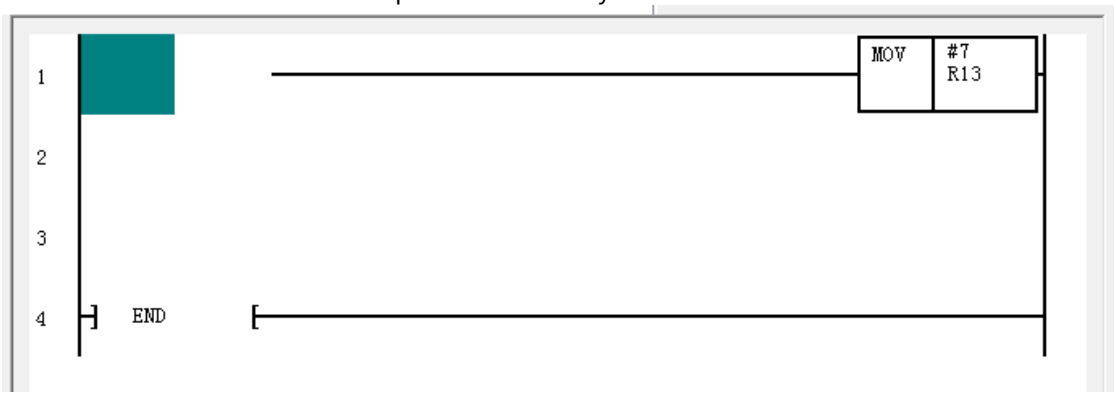


Cut Cell Copy Cell Paste Cell Delete Cell

- Move cursor to the cell

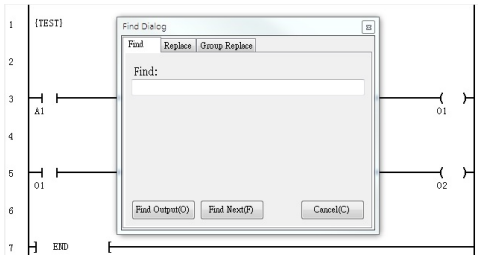
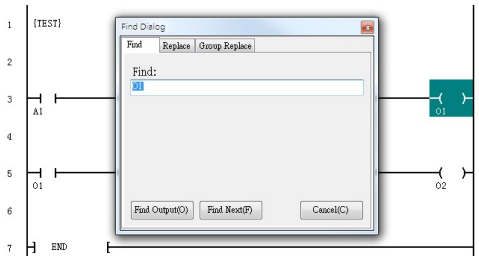
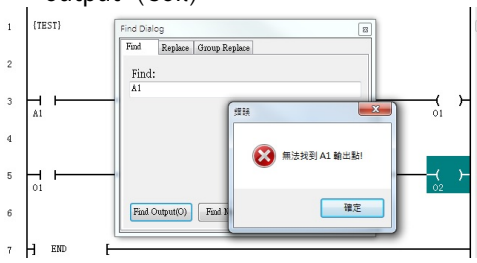
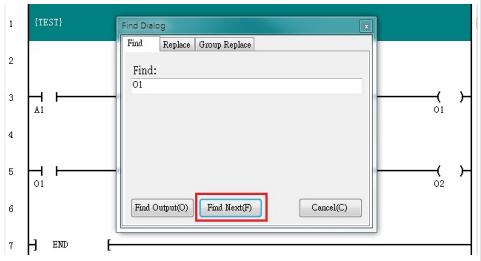


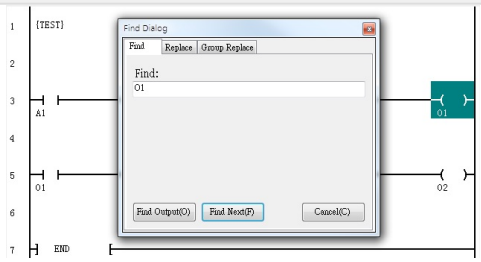
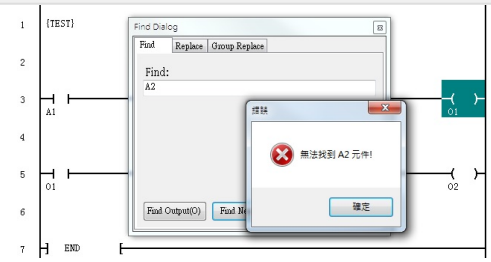
- Click "Delete Cell" in the "Edit" or press Delete on keyboard to delete the cell.



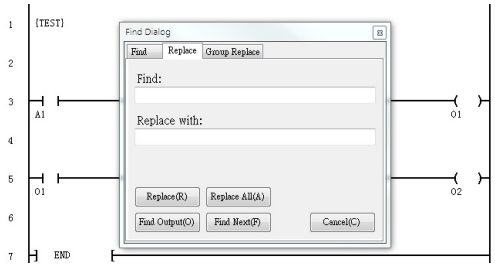
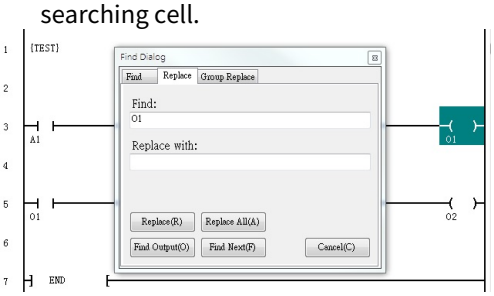
Find/Replace Cell / Goto Line

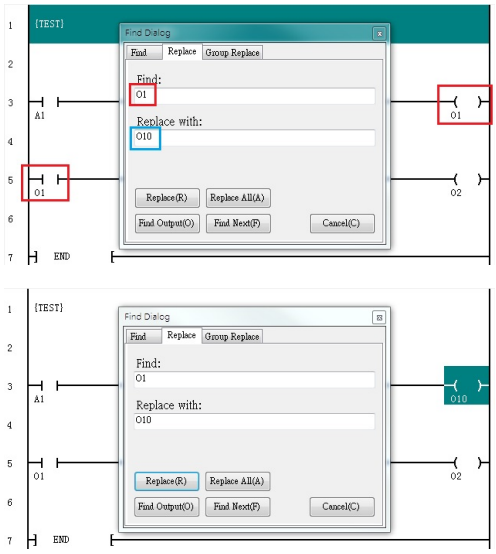
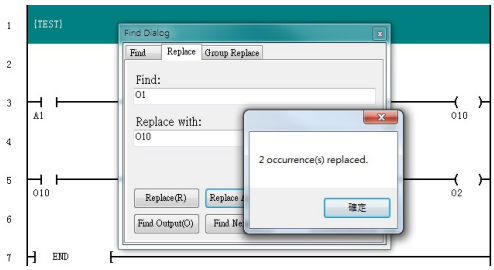
Find Cell Replace Cell Group Replace Goto Line

Operation	Figure	Note
Click "Find" in the "Edit" or press Ctrl+F on keyboard to find the cell.		
Key in the name of the cell you're finding]		• If the cursor is moved to a cell when applying the finding function, the cell's name will be taken as the default searching cell. • If the cursor is moved to a cell, the cell will become the target when applying "Find next" （or press Ctrl+F3） action.
Find Output Key in the name of the cell you're finding and click "Find Output", it'll find and move the cursor to the output （coil） cell that matches the name.		• If no cell matches the condition, hint message "Can't find XXX output" will pop-up and the cursor won't move. • For example：A1 is not an output（Coil） 
Find Next		• If no cell matches the condition, hint message "Can't find XXX" will pop-up and the cursor won't move. • For example：A2 does not exist

Operation	Figure	Note
Key in the name of the cell you're finding and click "Find Next", it'll find and move the cursor to the first cell which matches the name and also locates after the current cursor location.		

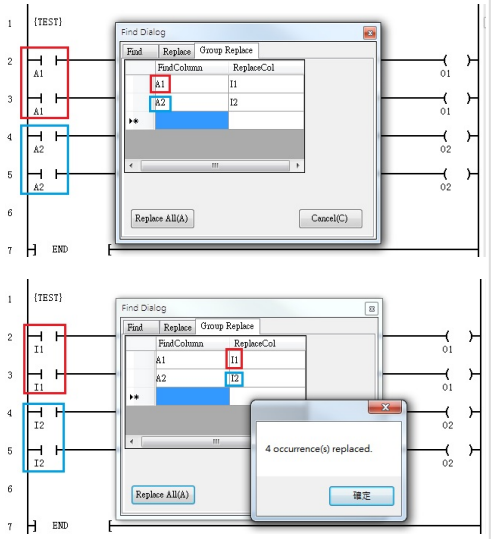
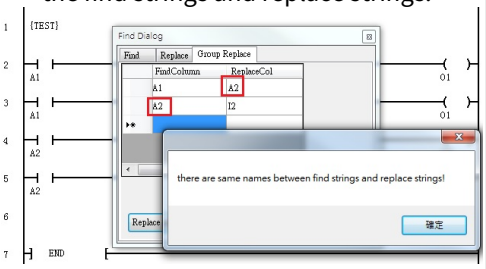
Find Cell | Replace Cell | Group Replace | Goto Line

Operation	Figure	Note
Click "Replace" in the "Edit" above or press Ctrl+R on keyboard can open the replace window		• If the cursor is moved to a cell when applying the replace function, the cell's name will be taken as the default searching cell. 

Operation	Figure	Note
Replace Key in the cell name which is going to be replaced and the new name then click "Replace". It'll find and move the cursor to the next cell which matches the name and replace it with the new name.		
Replace All		

Operation	Figure	Note
Key in the cell name which is going to be replaced and the new name then click "Replace All". It'll find all the cells that match the name and replace them with the new name. The cursor won't move in the action.		

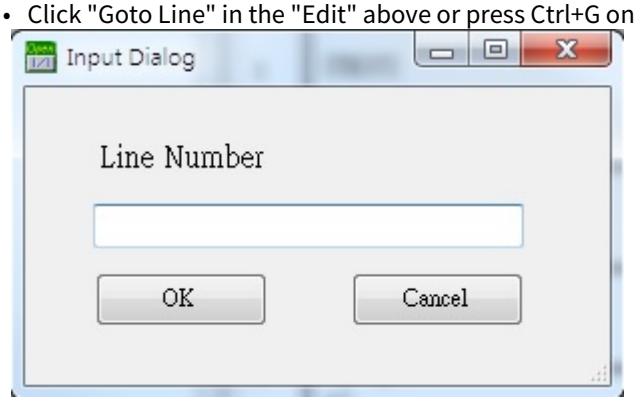
Find Cell Replace Cell Group Replace Goto Line

Operation	Figure	Note
Group Replace		• No same contents should be found in the find strings and replace strings. 

Operation	Figure	Note
Click "Group Replace" in the "Edit" to open the group replace window. It's able to key in multiple find strings and replace strings then execute at the same time.		

Find Cell Replace Cell Group Replace Goto Line

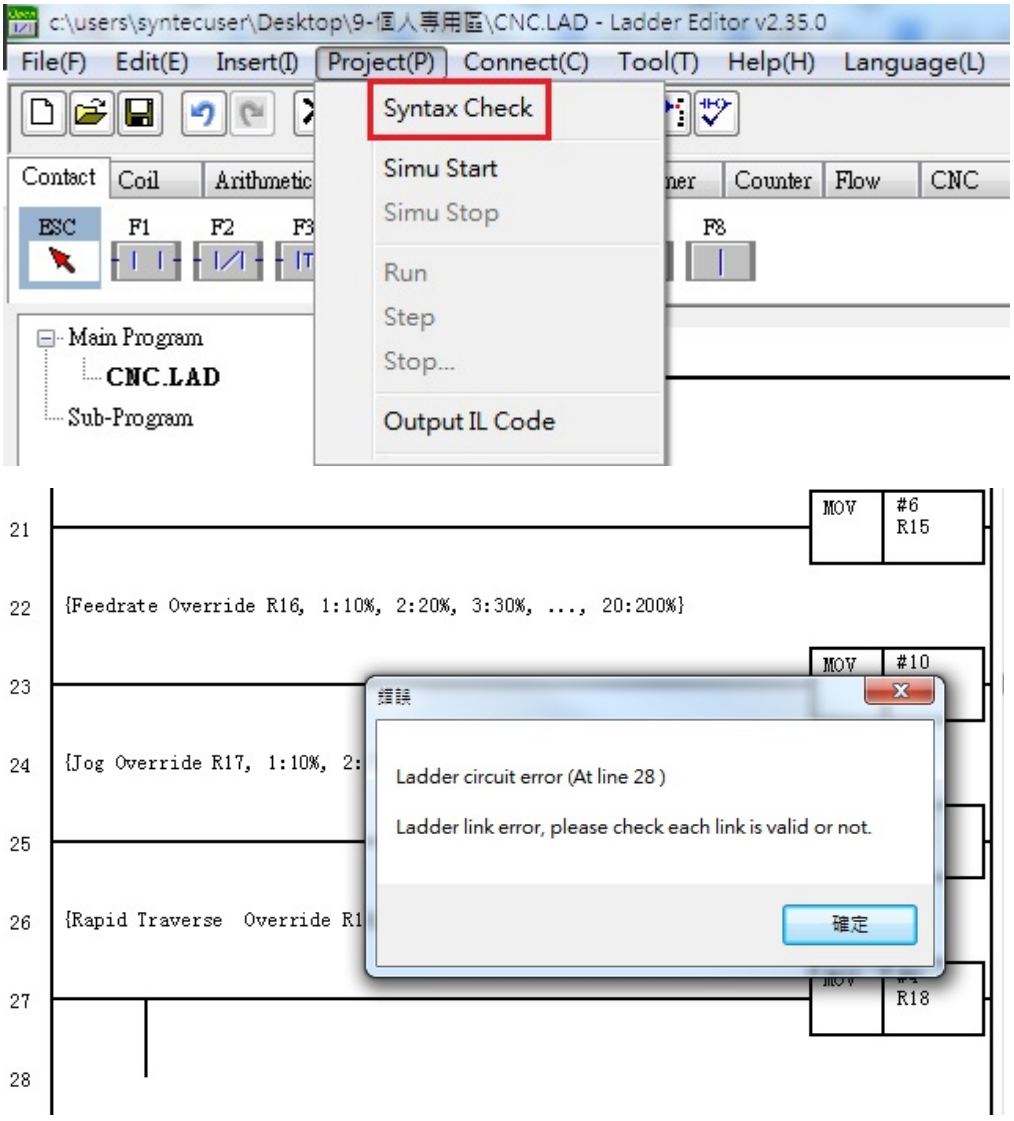
- Click "Goto Line" in the "Edit" above or press Ctrl+G on the keyboard can open the input window.



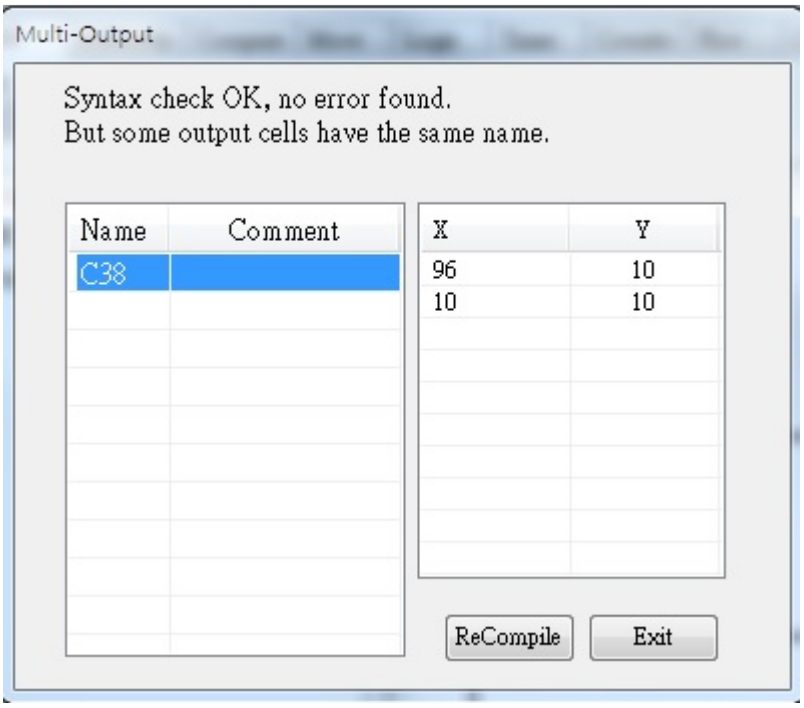
- Key in the line number and click OK to go to the assigned line.

Syntax Check

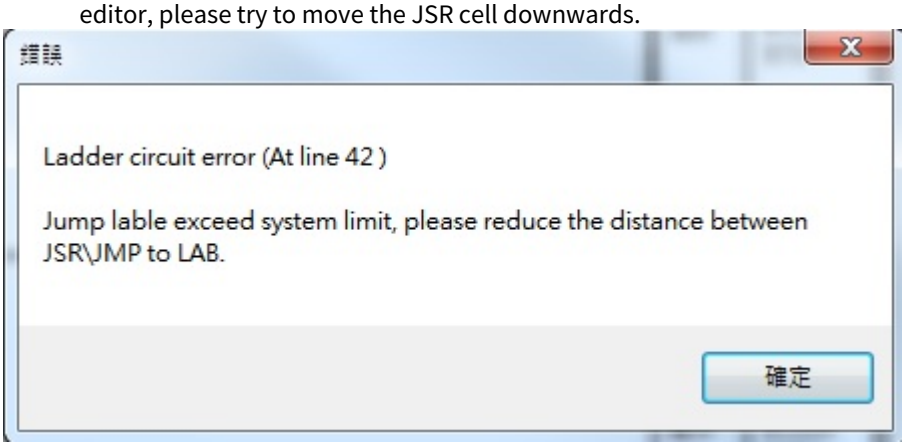
- The function is able to run syntax check for the ladder diagram in the command editing area：
  - If no errors are found, the cursor stays at the same location.
  - If error exists, the line number where error occurs will be shown and the cursor will be moved to the line also.



- If found repeating output, it'll list the name and address of all repeating output cells.



- If you see the message below, it means the compiling result of the contents exceeds the limit of PLC editor, please try to move the JSR cell downwards.

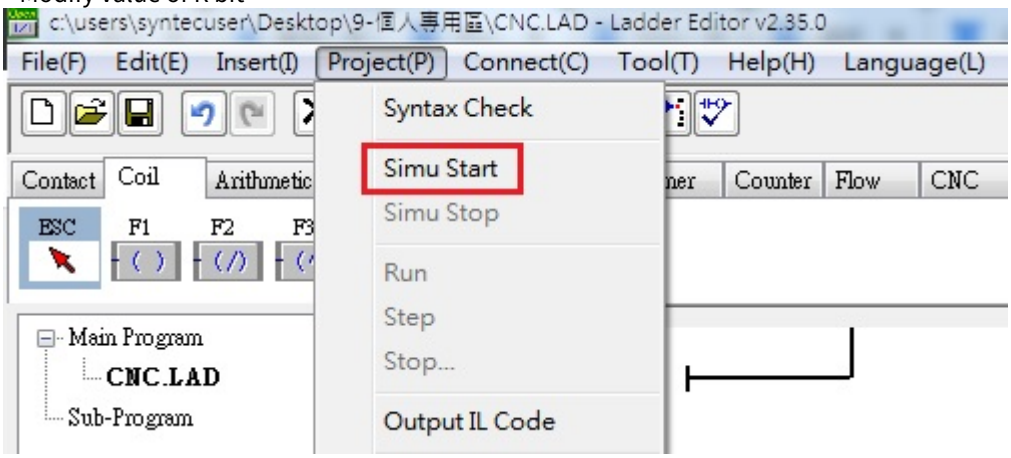


Simulation Function

The function can simulate the executing result of the current ladder diagram, also the status of the components will be shown simultaneously in command editing area and component list.

During the simulation, the actions below are supported：

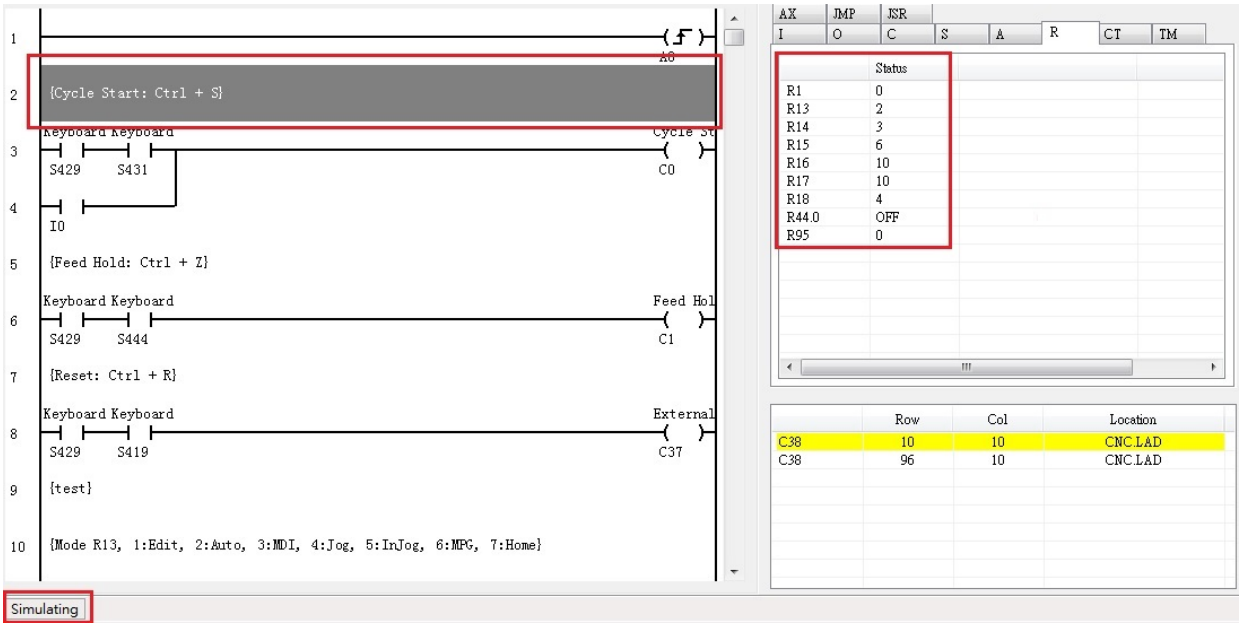
- Activate I, A, R bits instantly
- Update the status of PC keyboard related S bit instantly
- Modify value of R bit



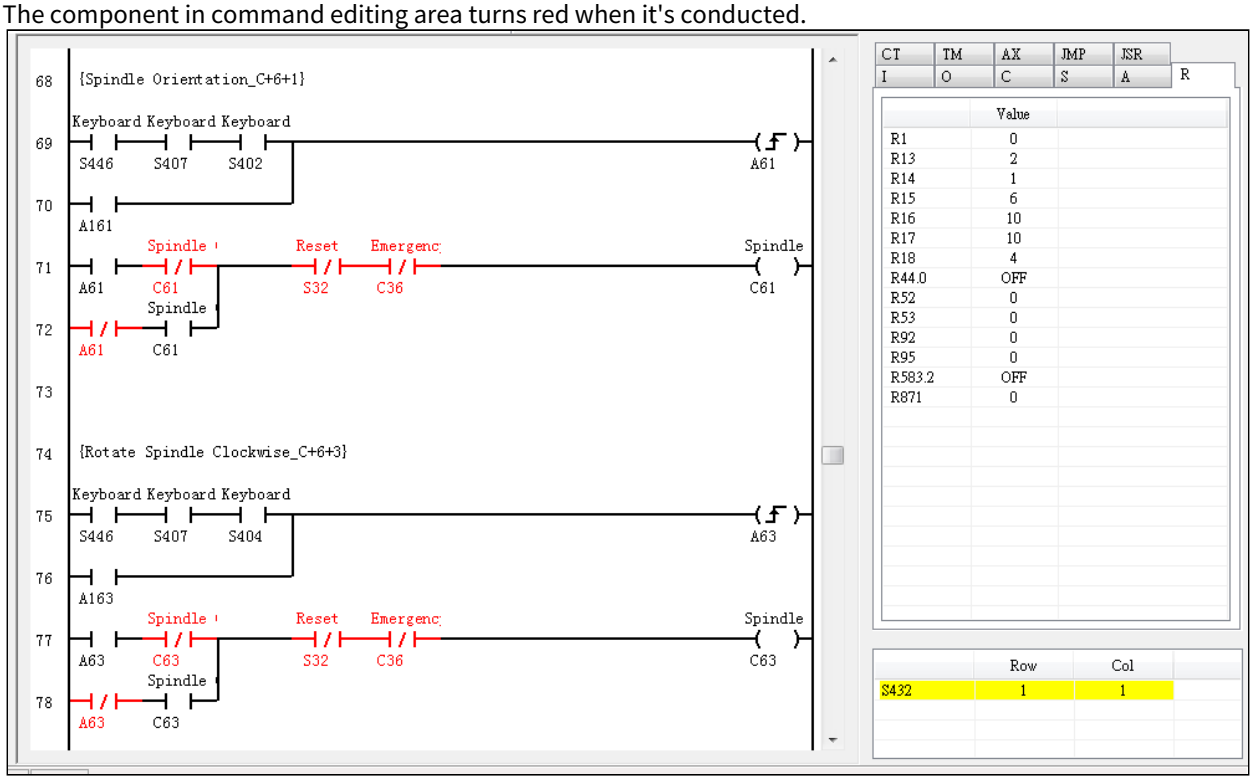
Simulation Interface

After the simulation has started, the status bar below will show "Simulating", the cursor color in command editing area turns from green to grey.

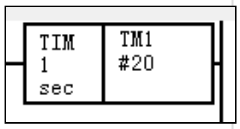
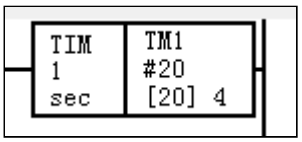
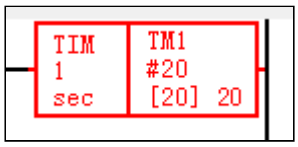
At the same time, the comment editing function of component list will be disabled and turns to show the status of various components including On/Off and values.



The component in command editing area turns red when it's conducted.

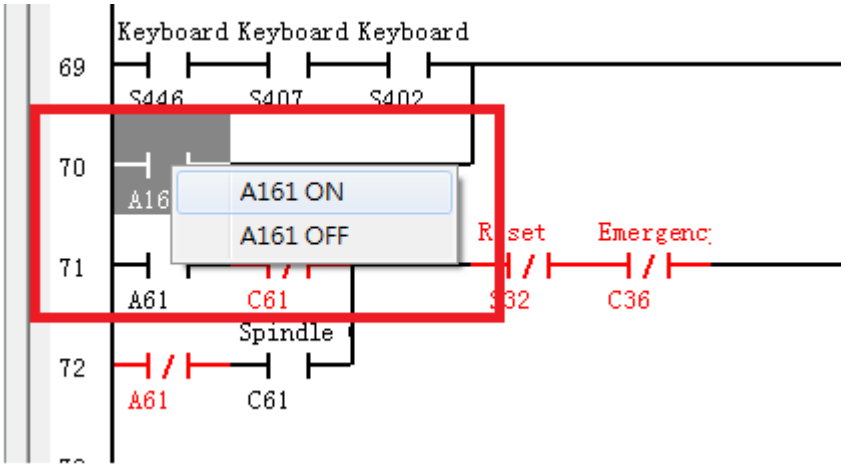


Timer and Counter components provides instant value update function.

Timer	Instant value update in simulation	Conducted status
		

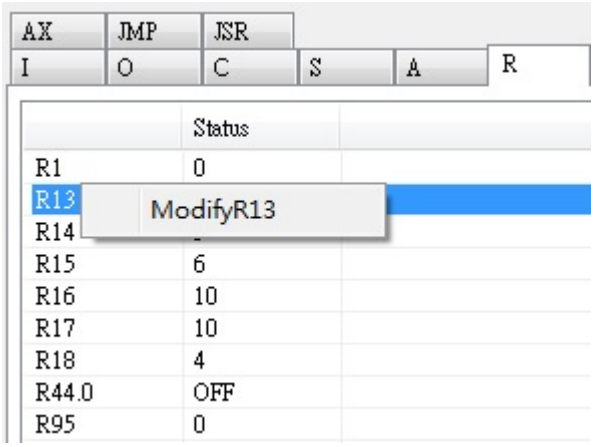
Activate I-bit, A-bit, R-bit

During simulation, the ON/OFF list will appear when right clicking on I-bit, A-bit, R-bit which you want to activate.

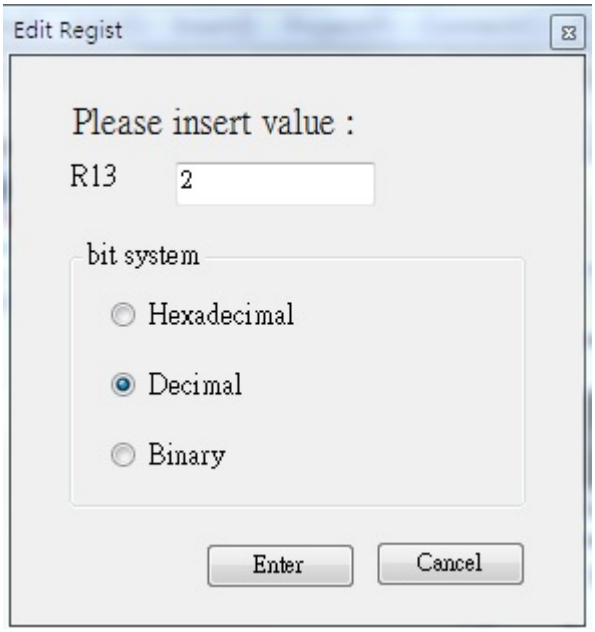


Modify R value

During simulation, it's also able to modify the R bit value by right clicking on the R bit on the component list.



There are total 3 input formats for R bit value modification：hexadecimal, decimal and binary.



Watch Cell List

After the simulation has started, the Watch Cell List will also pop-up.

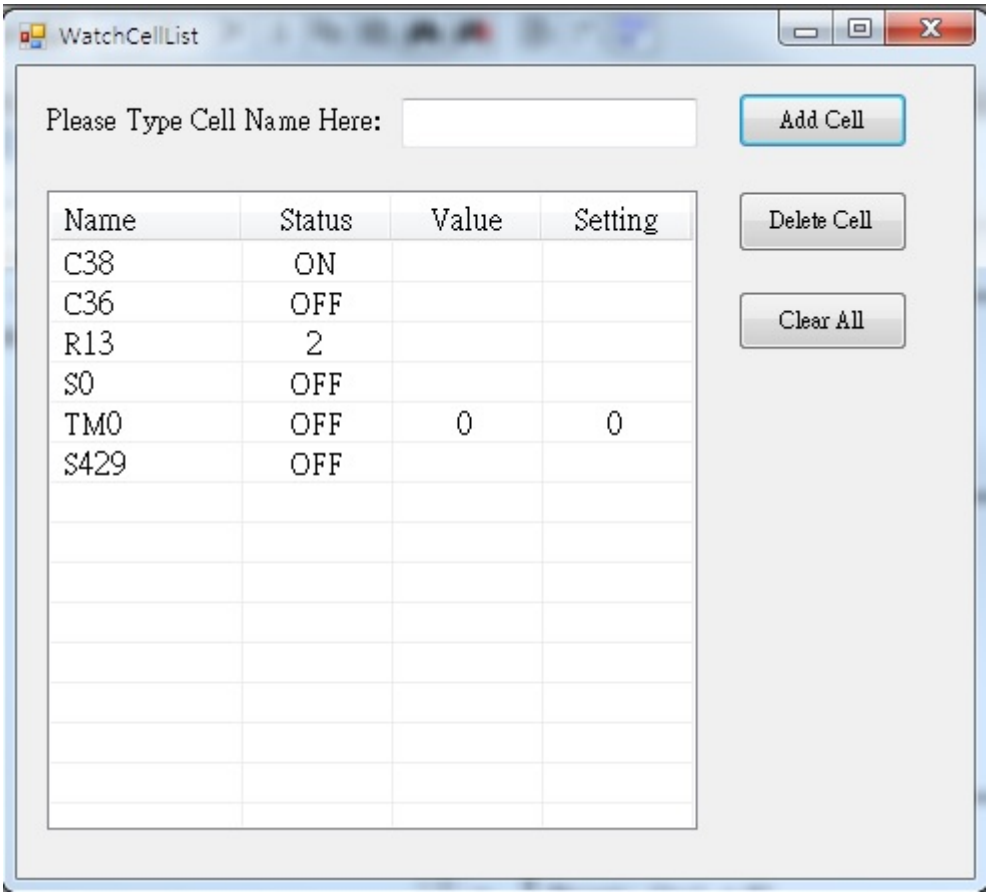
Key in the components you want to monitor into the inputbox, the list below will update instantly during the simulation.

Add Cell：Key in the component name into the inputbox and click "Add Cell" or press Enter.

Delete Cell：Click the component and click "Delete Cell". It's also able to delete the whole list.

Provided types：

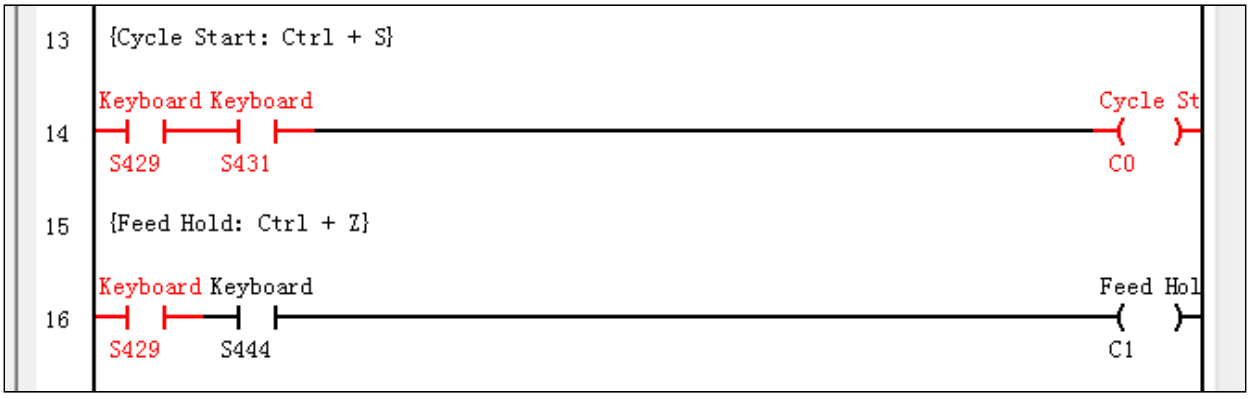
Component Type	I-Bit	O-Bit	C-Bit	S-Bit	A-Bit	Register	Timer	Counter



Simulation of S bit with keyboard

During simulation, it's able to activate the corresponding S bit with PC keyboard. (Note 2)

The example below is pressing Ctrl + S.



Notification

- Run syntax check before simulation, the simulation can't be executed if syntax errors found.
- Before simulation, the editor will ask whether to save or not if you have modified the ladder.

- During simulation, all the editing actions and file operations are invalid.
- The values of Timer and Counter components on the watch list will only be update simultaneously after they're activated.

Note 1：Components supporting the simulation function

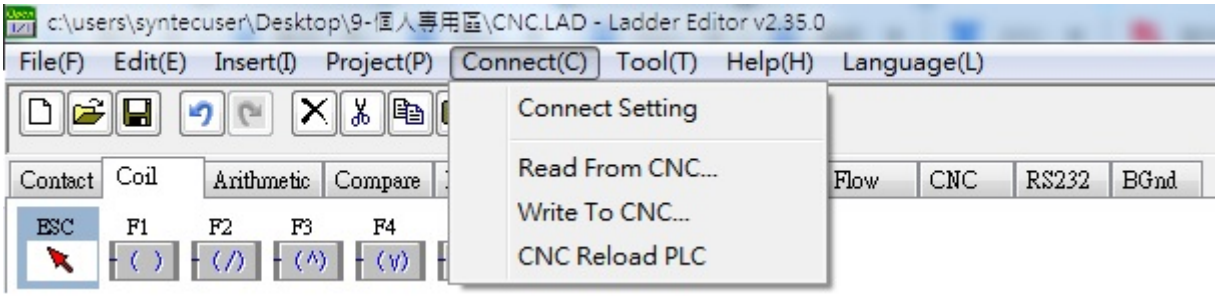
Type	**Items**							
Contact	-| |-	-| /|-	-|T |-	-|T/|-	-|C |-	-|C/|-		
Coil	-( )-	-(/)-						
Timer	tim1ms TIM, #	tim10ms TIM, #	tim100ms TIM, #	tim1s TIM, #	tim1ms TIM, R	tim10ms TIM, R	tim100ms TIM, R	tim1s TIM, R
Counter	cntUp CT, #	cntDn CT, #	cntRingUp #	cntRingDn #	cntUp R	cntDn R	cntRingUp R	cntRingDn R
Comparing	CMP > R, R	CMP < R, R	CMP = R, R	CMP > #, R	CMP < #, R	CMP = #, R		

Note 2：PC keyboard corresponding S bit

Esc 401	F1 458	F2 460	F3 461	F4 462	F5 463	F6 464	F7 465	F8 466	F9 467	F10 468	F11 469	F12 470		
` 441	1 402	2 403	3 404	4 405	5 406	6 407	7 408	8 409	9 410	0 411	- 412	= 413	\ 443	Back 414
Tab 415	Q 416	W 417	E 418	R 419	T 420	Y 421	U 422	I 423	O 424	P 425	[426	] 427	Enter 428	
Caps 458	A 430	S 431	D 432	F 433	G 434	H 435	J 436	K 437	L 438	; 439	' 440			
Shift	Z 444	X 445	C 446	V 447	B 448	N 449	M 450	, 451	. 452	/ 453	Shift 454			
Ctrl 429		Alt 456		Space 457								Alt 456		Ctrl 429

Print	Scroll 470	Pause		Num 469	/ 453	* 455	– 474
Insert 482	Home 471	Up 473		7 Home 471	8 ↑ 472	9 Up 473	+ 478
Delete 483	End 479	Down 481		4 ← 475	5 476	6 → 477	
	↑ 472			1 End 479	2 ↓ 480	3 Down 481	Enter 428
← 475	↓ 480	→ 477		0 482		. 483	

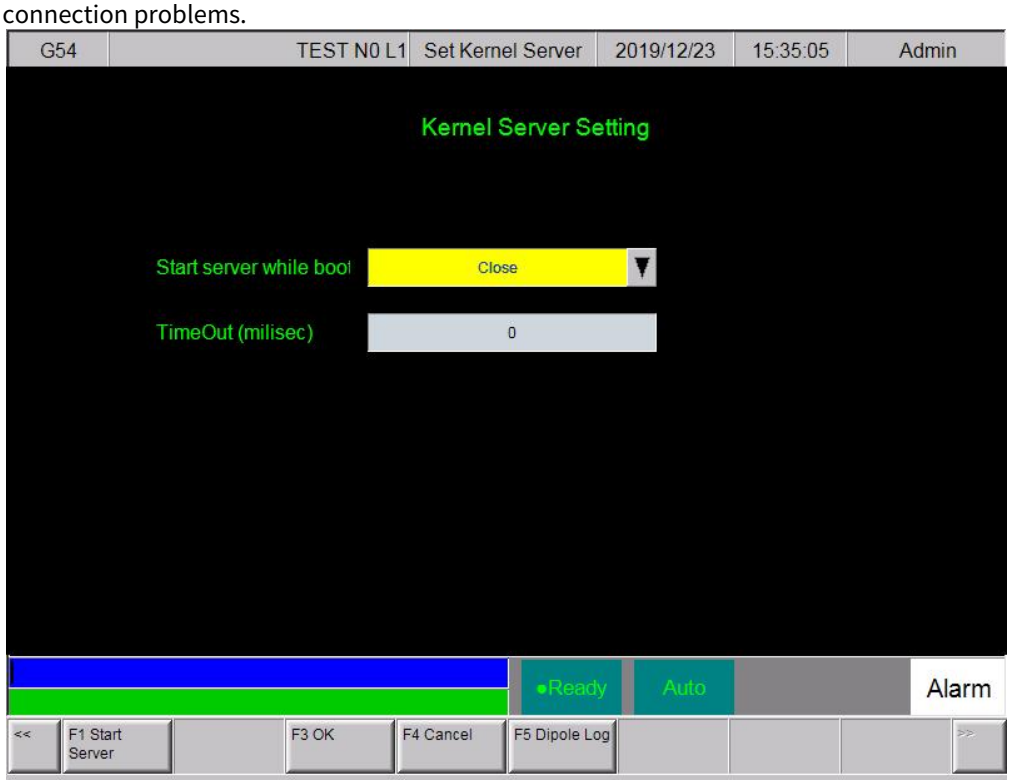
Connection Operation



1. Software version table of PLC editor and controller：

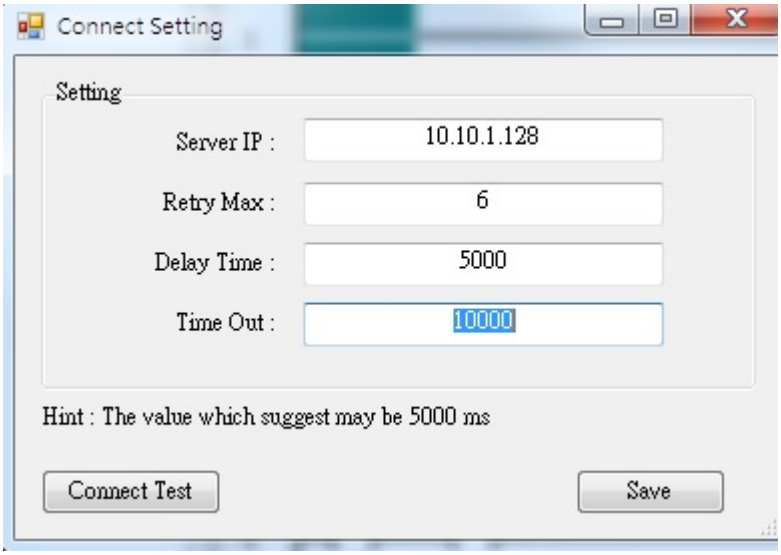
PLC editor	Controller
After 2.29.4	After 10.116.36B, 10.116.37 and 10.117.37
Before 2.29.3	Before 10.116.36A, 10.116.36 and 10.117.36

2. Please make sure the controller and PC are connecting successfully before applying the function. Reference：Instruction of Network Setting
3. After finish the setups above, please start the server on the controller.
4. For server setups, it suggests to set the timeout to be 0 to avoid unstable internet speed and connection problems.



PC Connection Settings

Connection setting of PC to connect with controller



Setup Option	Recommend Value
Sever IP	IP address of the controller
Retry times	5
Retry delay time	3000 ms
Timeout	5000 ms

Setup Procedure：

1. Set up the 4 connection information above
2. Click "Connection test"
3. Make sure connection is successful.（Refer to the hint shown bottom left）
4. Click "Save"

Read from Controller

Download the ladder diagram of the controller to PC.

Operation Procedure：

1. Select the saving location and file name of the ladder diagram.
2. Confirm the download result, hint window will pop-up after the download is finish.
3. After download finish, PLC editor will load the ladder diagram automatically.

Write to Controller

Upload the current ladder diagram to the controller.

1. The file will be saved before upload, users will be asked to select the saving path if it's a new file.
2. Suggest running syntax check before upload to avoid errors after writing to the controller, it might cause boot failure of the controller.
3. Confirm the upload result, hint window will pop-up after the upload is finish.
4. After upload successfully, reboot the controller to read the new ladder diagram.

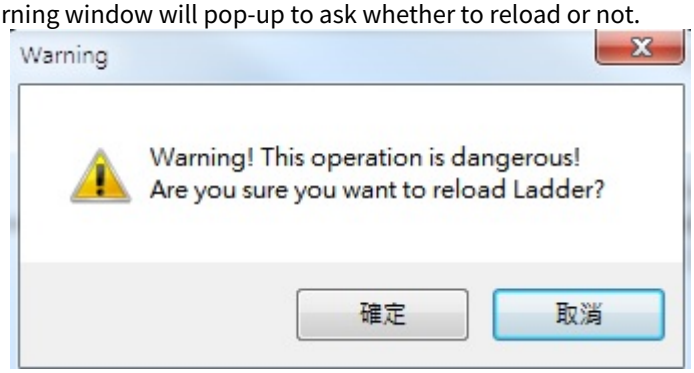
CNC Reload PLC

PLC editor supporting version：2.31.0

Controller supporting version：After 10.116.45（included）

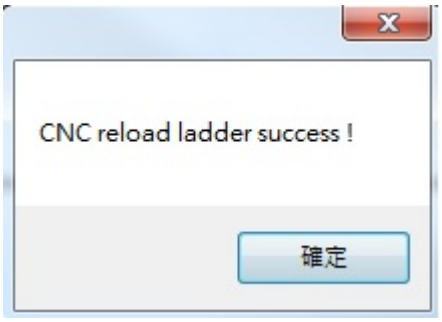
This function is able to make the controller reload the ladder diagram without reboot, but the function is only valid in unready state.

1. Warning window will pop-up to ask whether to reload or not.

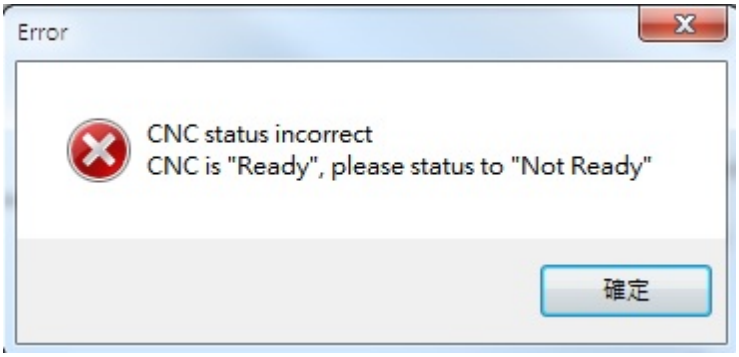


2. Status report

Success：



Fail：

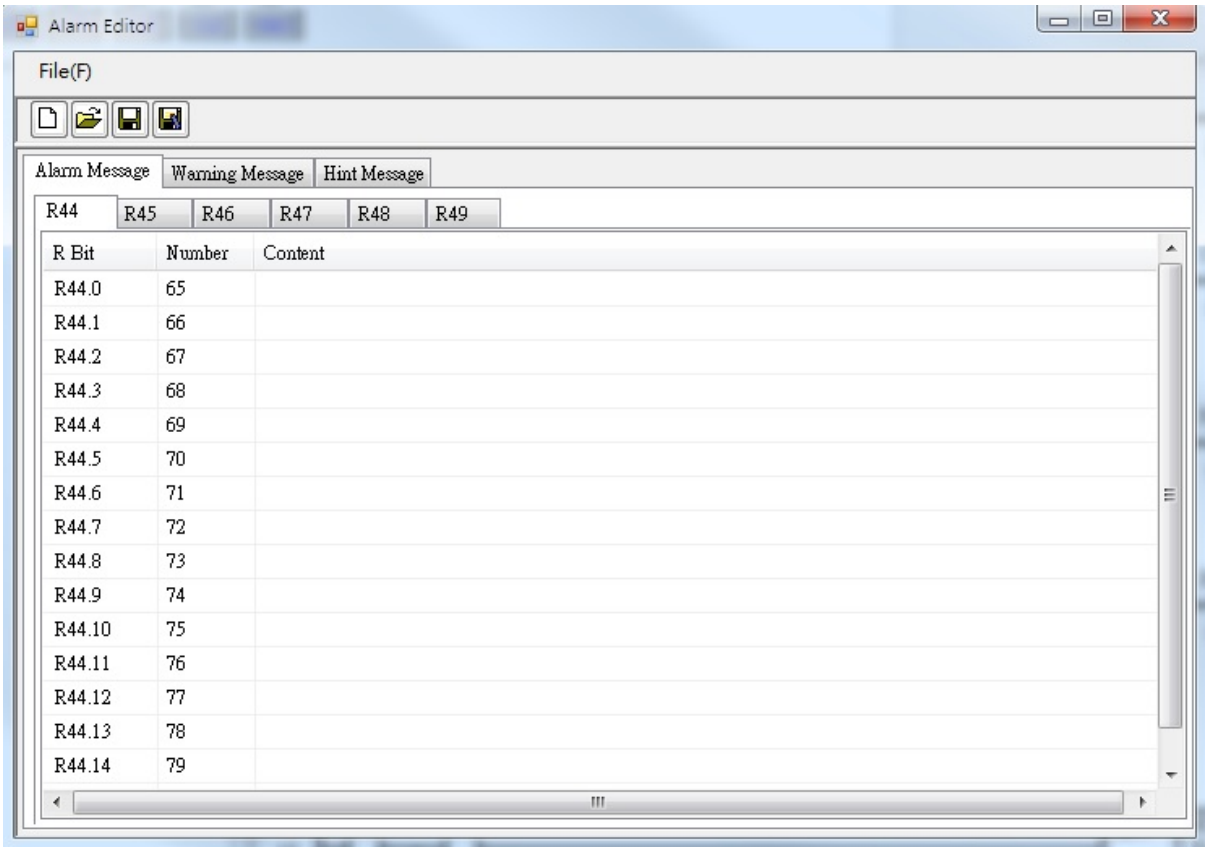


User-defined Alarm Edit

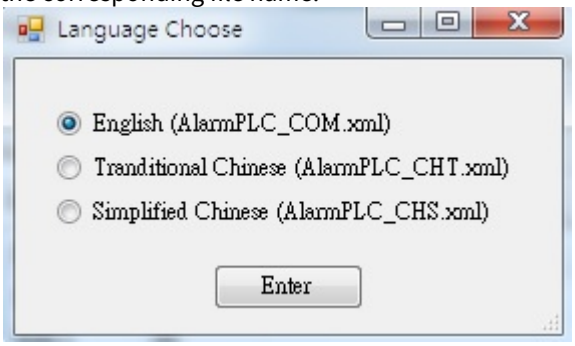
Supporting version：2.20.0

After PLC editor is opened, click the drop-down list："Tool"→"Alarm Editor", open the alarm editor interface, shown as below：

Currently R44~R49 are used for user-defined alarm, you can find the alarm string numbers on the alarm editor.



The languages must be selected first when saving the user-defined alarm files, PLC editor will then generate the corresponding file name.

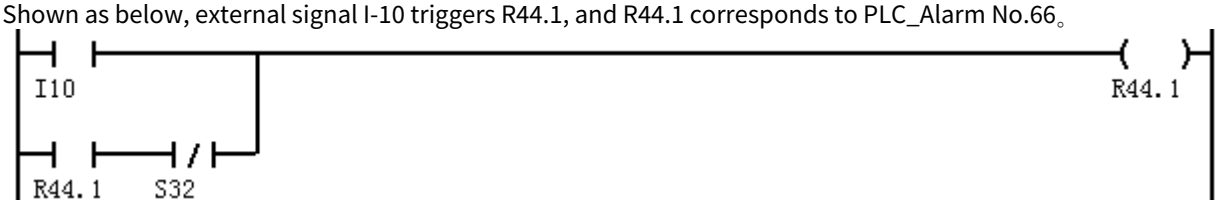


Notes：

1. The str file for DOS can be saved as xml or str file; but the xml file for CE can only be saved as xml file.
2. When opening old alarm files of CE, the file name can only be AlarmPLC_CHT.xml / AlarmPLC_CHS.xml / AlarmPLC_Com.xml.

To Trigger User-defined Alarm

Shown as below, external signal I-10 triggers R44.1, and R44.1 corresponds to PLC_Alarm No.66。



Switch Language

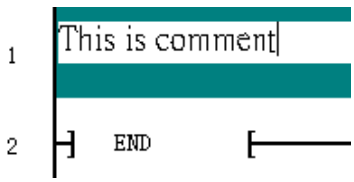
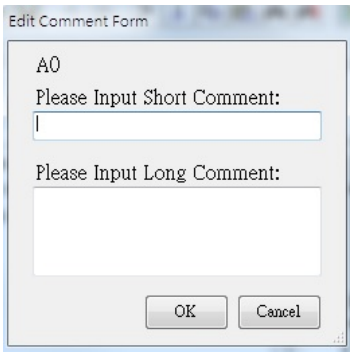
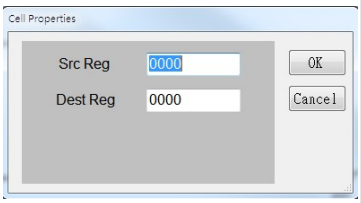
The function is able to switch the language of the interface.

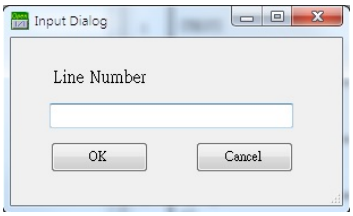
Click "Language" on the drop-down list, select the switching language and restart the PLC editor.

Now provides：Traditional Chinese, Simplified Chinese, English

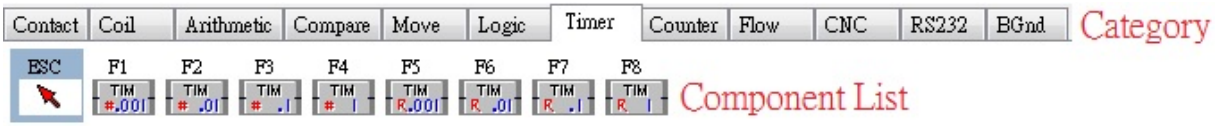
Hotkeys

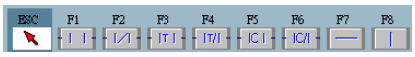
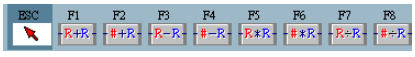
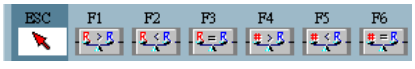
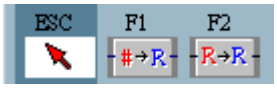
Interface Operation

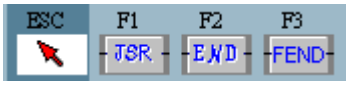
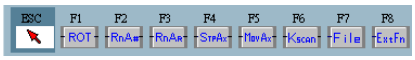
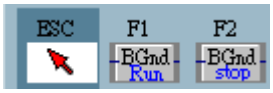
Function	Hotkey	Operation Example	Note
Insert Comment	Ctrl + M		The comment will be inserted above the current cursor location.
Comment Edit Window	Ctrl + Enter		The cursor must be moved to the component you want to edit.
Cell Property	Enter		The cursor must be moved to the component you want to edit.
Move cursor location	Arrow keys		The cursor must be in the editing area.
Add new line	Ctrl + L		The new line will be added above the current cursor location.

Function	Hotkey	Operation Example	Note
Go to specific line	Ctrl + G		

Component Switch



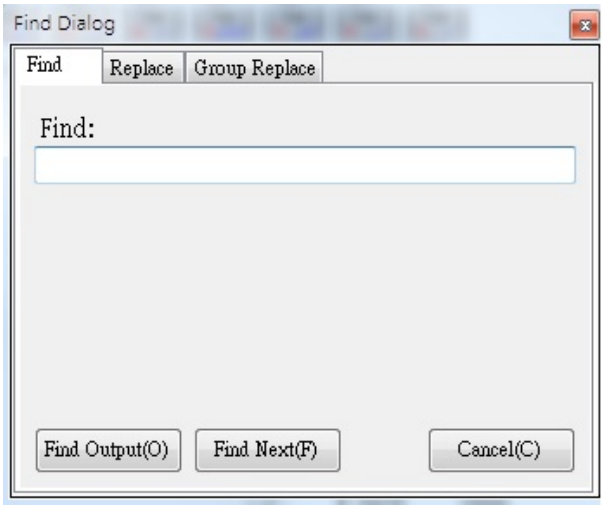
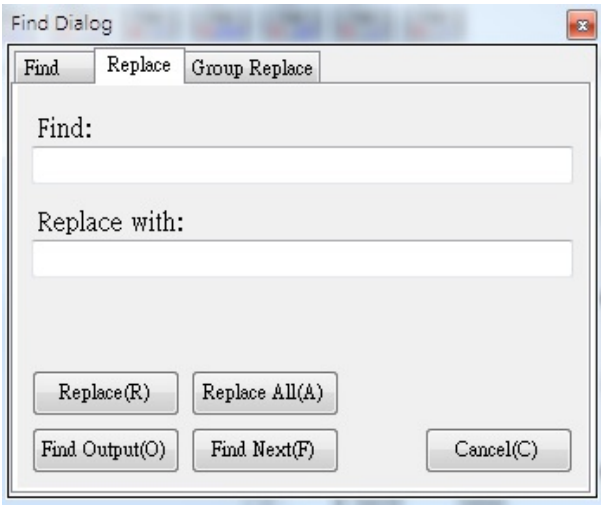
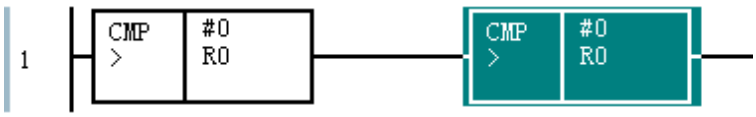
Function	Hotkey	Object list corresponding to the label
Switch Category	Ctrl + Left/Right	Please refer to the picture above.
Switch to specific category	Ctrl + Number Key	Please refer to the picture above.
Switch component under category	Esc, F1~F10	Please refer to the picture above.
Switch category to Contact	Ctrl + 1	
Switch category to Coil	Ctrl + 2	
Switch category to Arithmetic	Ctrl + 3	
Switch category to Compare	Ctrl + 4	
Switch category to Move	Ctrl + 5	

Function	Hotkey	Object list corresponding to the label
Switch category to Logic	Ctrl + 6	
Switch category to Timer	Ctrl + 7	
Switch category to Counter	Ctrl + 8	
Switch category to Flow	Ctrl + 9	
CNC category	Can only be switched with arrow keys currently	
RS232 category	Can only be switched with arrow keys currently	
BGnd category	Can only be switched with arrow keys currently	

File Related

Function	Hotkey
New file	Ctrl + N
Open file	Ctrl + O
Save file	Ctrl + S

Viewing Related

Function	Hotkey	Operation Example
Find	Ctrl + F	
Replace	Ctrl + R	
Find Next	Ctrl + F3	If the cursor is on a component, it'll jump to the next identical component. Example：The cursor is on CMP #0  After pressing Ctrl + F3, the cursor move to next CMP #0. 

Program Editing

Function	Hotkey
Cut	Ctrl + X
Copy	Ctrl + C
Paste	Ctrl + V
Delete	Delete
Undo	Ctrl + Z
Redo	Ctrl + Y

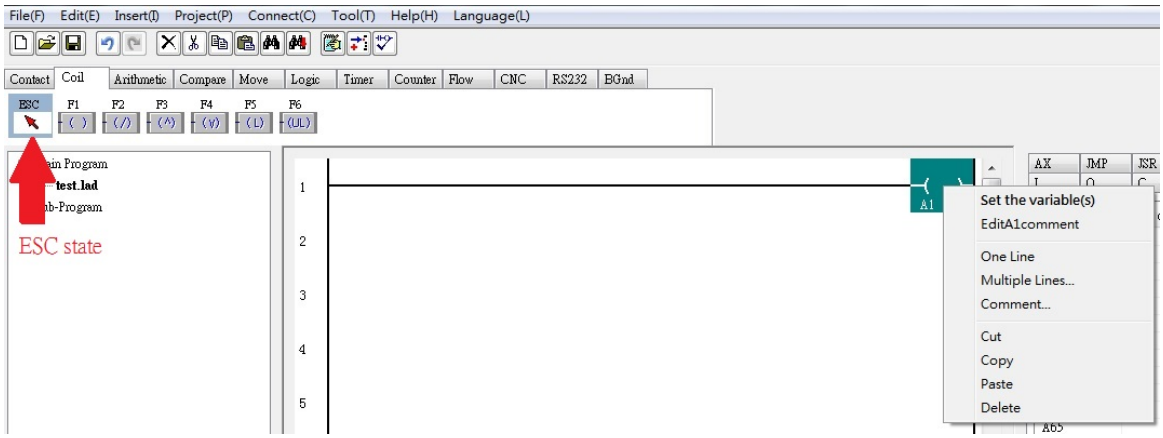
Operation Tips

Command editing area

There are 2 situations when right click on the command editing area：

- If the component bar is in "Non-ESC state", it'll return to "ESC state" after right click, which means no component is selected.

- If the component bar is already in "ESC state", the function list will pop-up after right click.



Component Index

When user selects any component in "Command Editing Area" or "Component List", the yellow cursor in "Component Index" will show the selected component.

- Double click on the component in component index, the cursor in editing area will jump to the selected component, but the yellow cursor in component index won't move.
  - Supporting version: before 2.26.0 and after 2.31.4
- Double click on the component in component index, both the cursor in editing area and the yellow cursor in component index will jump to the selected component.
  - Supporting version: 2.26.1 ~ 2.31.3