PLC overview

INOVANCE PLC (Programmable Logic Controller) is an electronic system with digital calculation operation, which is designed for applications in an industrial environment. It reads external input state signals of keys, sensors, switches, and pulse waves. Based on these input signal states or values and the internal storage pre-prepared program, the logic, sequence, timing, counting and arithmetic operations are implemented by micro-processor, then producing corresponding output signals, such as: switching relay, and controlling machinery equipment operation. The settings of the program and monitor device can be easily edited and modified by using computer or program editors, to maintain the on-site program or to debug.

Fundamental controlling principle

PLC Operation principle:

Programmable Logic Controller (PLC) uses loop-scanning operation process to perform tasks such as input point scanning, user program executions, output point regenerations, and internal and communication processing.

Prior to the operation of the PLC, the control logic between input point and output point could be programmed with computer software and be downloaded to PLC. When the PLC is executing the commands, input point signals will first be scanned and imported into the PLC. The calculation and logic processing will then be completed according to the controlling procedure. The results will change the value of output point and convert the value into electric output signals to control the operations of various equipments.

The PLC uses loop-scanning operating method to perform constant and repeated execution input point scanning, user program executing, and output point regeneration to achieve the purpose of complete monitoring and controlling of the equipments.

User program control principle:

The input point of a PLC is also known as a "contact point" in the user program. It has the same functionality as the switch contact in industrial equipments, representing the conducting or shutdown state of the power flow.

The input point is saved as a soft component in a PLC. When input point is at high electrical level, the corresponding soft component will be

in conducting state and will be included in user program's logic calculation to further influence the value of output point; the output point is also called "coil", representing the conducting or shutdown state of the power flow. The value of the corresponding output point soft component is determined by the input points and the control logic's calculation results.

When output is being regenerated, the soft component value is converted into electrical signals and being released from the output transistor or relay, and further to accomplish the controlling of equipments.

Programming language

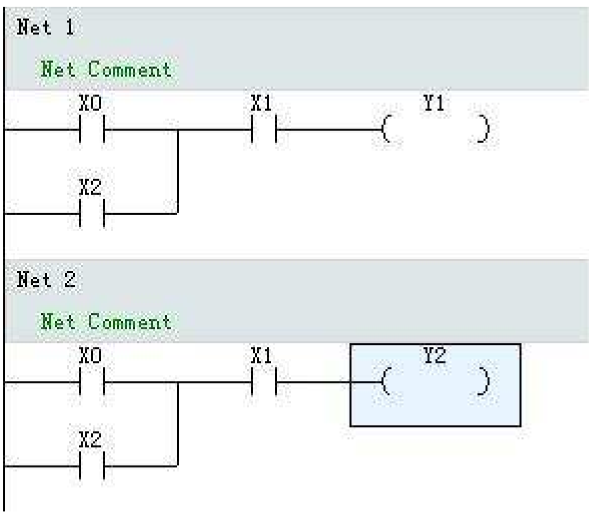
INOVANCE Autoshop software programmable software supports three commonly used languages: Ladder Diagram (LD), Instruction List (LI) and Sequential Function Chart (SFC).

The main program could be written in any one of these three program languages, but for sub-program and interrupting program only the LD and IL can be used. In addition, the built-in program in sequential function chart can only be programed using LD.

LD programming:

Based on the model of relay control system, PLC ladder diagram programming method uses the electrical theory and adopts the components used in the designation similar to the time electrical device, such as button X, intermediate relay M, time relay T, counter C, contact points, and etc..

Illustration:



In the above ladder diagram, the execution order is calculated gradually using the user's program network as a unit. The "network"

refers to a set of component block that is with networking relevance. Please see the two networks illustrated above.

The calculation implementation starts from the first network, and then it moves onto the next network until the final network.

For each network, the calculation is implemented from left to right, in which the contact statuses of the components are logically synthesized one by one, until the end. After that the result will be either exported to the "coil" of the component, or to the logic controller to determine the execution of an operation.

As illustrated above, the specific implementation logic of each network is shown as the following:
1. The input point X0 value is first loaded as the current value, and then the input point value X2 will also be loaded;

2. After choosing "or" calculating the above two values, the result becomes the current value;

3. Load and calculate the input value X1 and the current value;

4. The result determines the conduction flow at the final controller output point Y0.

Instruction list programming:

The instruction list program editor is a text editor. All logic and calculations are inputted using instruction and operand. Based on the functional instructions completed and the associated soft components in operand, the value of soft component is read, and logic processing and the value of soft components are written.

Illustration:

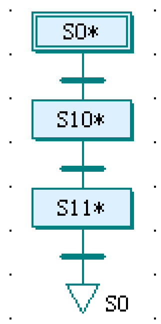
```
//Network 1 program comments
Instruction      Soft component

LD         X0
OR         X2
AND        X1
OUT        Y1

//Network 2 program comments
Instruction      Soft component
LD         X0
OR         X2
```

AND X1
OUT Y2

Sequential function chart programming:

Sequential function chart is a programming language that divides the operating flow or procedure of equipments into a number of conversions between operation steps.
A standard sequential function chart consists the conversion requirement, jump, and reset in initial step, general step, between steps. Each step is a processing procedure of mechanical equipments. In one step it can consist built-in ladder diagram, which is the processing procedure that requires to be completed in that step.Conversion requirement refers to the activation requirement between the completion of one procedure and the beginning of next procedure. It also requires built-in ladder diagram to indicate the conversion requirement.

Illustration:



As illustrated in the above sequential function chart, the initial state is S0. When the conversion requirements from S0 to S10 are not satisfied, S0 procedure will always be in execution mode. Once the conversion requirement is satisfied, S0 will be terminated and S10 will be executed, and so on and so forth. When S11 is completed at the end and the reset requirement from S11 to S0 is satisfied, S11 procedure will be terminated, and the initial step of S0 will be restarted.

Program conversion:

These three programming languages mentioned above are interchangeable based on user's preference or the practicability of the controlling environment. Users can choose the most appropriate programming language.Due to the unique nature of sequential function chart language, the ladder diagram or instruction list which must be programmed to comply with the programming syntax of the sequential function chart so that it can be correctly converted to sequential function chart.

Quick start

The main contents in the Quick Start include:

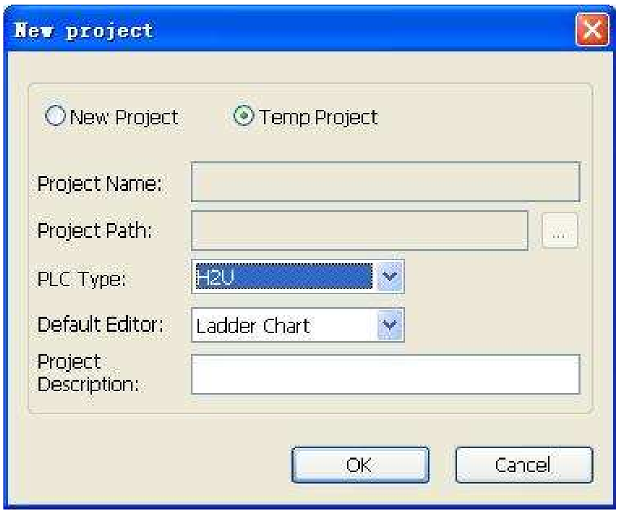
1) How to download a program to PLC via Autoshop and run it.

2) The descriptions for all interface modules in the Autoshop software.

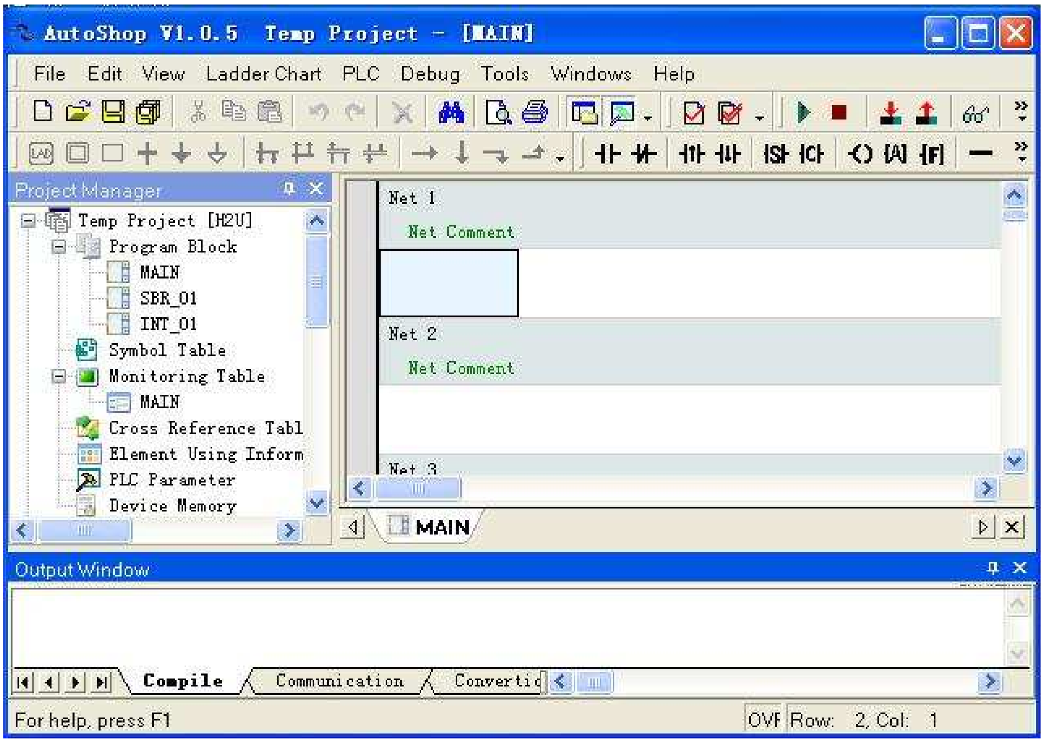
3) The description for all functional modules in the Autoshop, including the newly created projects.

Create project

The project should be initially created for the program after startup of the programming environment. Click " file"---"new project" and the following dialogue will be displayed:



In this example,two mode which is named create project and create temp project can be choosed(When choosed create project mode user must set project name,project path and other item.But when choosed create temp project mode,project name and path auto created by system,it is useful when testing,you can also save project as your mind).H2U should be chosen as the PLC type and the default editor is the ladder diagram. Click "OK" after selection and a new project will be created and the main routine will be opened in the edit mode as a default, as shown in following figure:



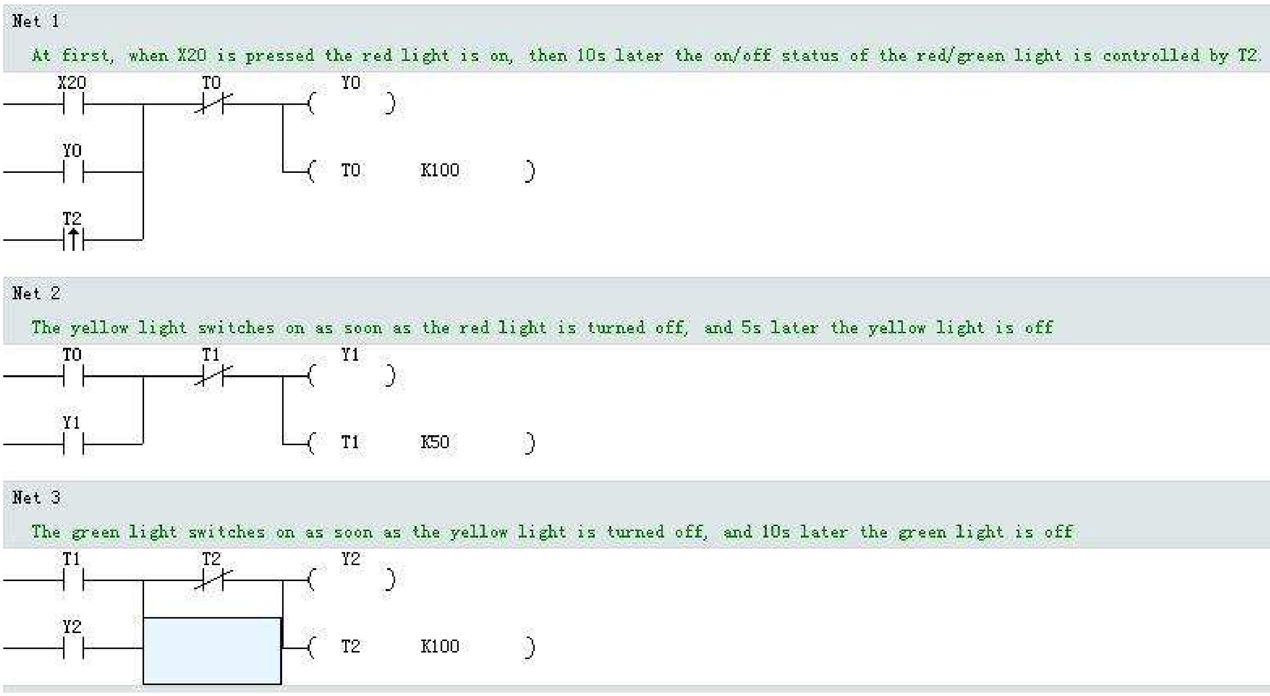
Please refer to pertinent chapters for detailed project management operations.

Write a simple ladder diagram program

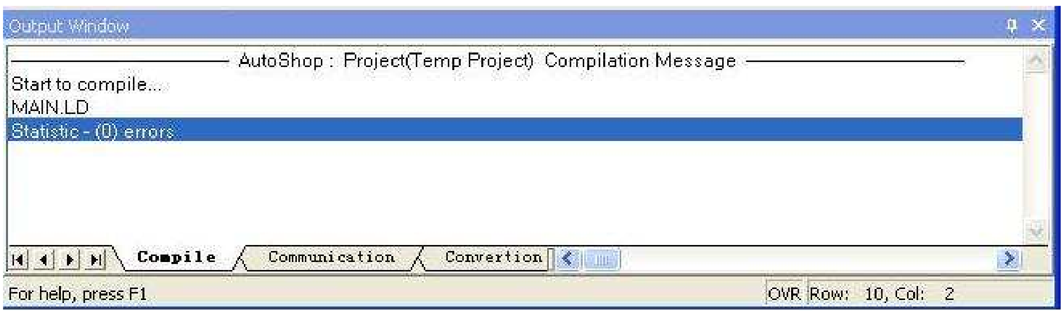
write a simple ladder diagram program:
In this example we will write a traffic light time control program in which component Y0, Y1 and Y2 represent red, yellow and green light respectively. The control logic is described as following:

First the red light illuminates for 10 seconds and then it goes out. Then the yellow light illuminates for 5 seconds and then it goes out. Then the green light illuminates for 10 seconds and then it goes out too. Then the red light illuminates again and the above process repeats. (The accuracy of timers T0, T1 and T2 is 100ms in this routine)

Following is the ladder diagram program in the Autoshop programming environment:

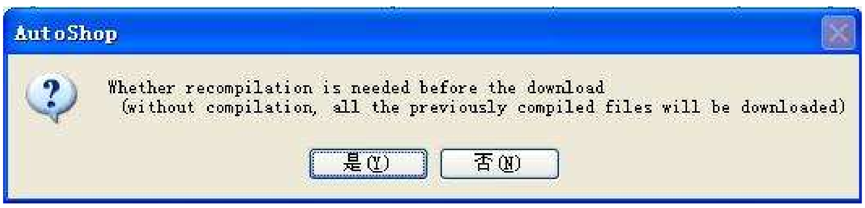


When the program is completed you can save the current project to your PC just by clicking the menu "File" - "Save project". After the storage you need to check the current program has no error and then compile it to object file which can be downloaded into the PLC. After clicking the menu "PLC" - "compile all" the system will compile all the current program while showing compile result in the output dialogue. If there is no error then following text will be displayed:

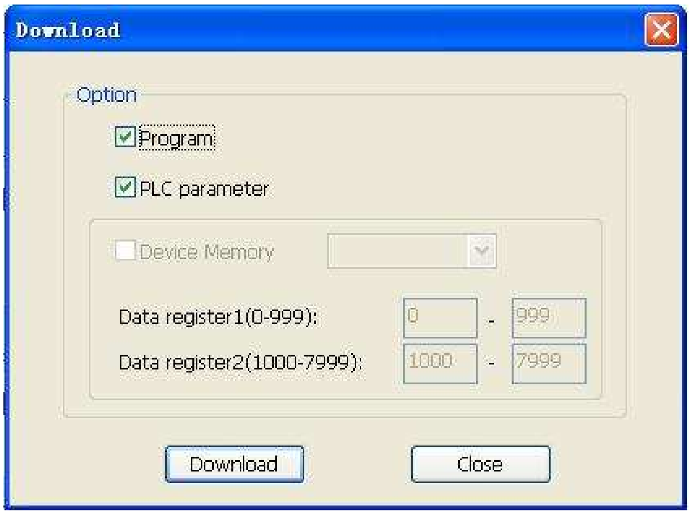


Program download

Click on the DOWNLOAD menu item in the PLC menu to download the program. Firstly, the user will be asked that if a recompilation is required. If you choose not to recompile, the file you will download is the very file of last compilation. The prompt dialog box should be as follow:



If you click on YES, the program will be recompiled; and if you click on NO, the file of last compilation will be kept and a dialog box as follow will pop up:



Select items to be downloaded in DOWNLOAD OPTIONS as required. Since only the written program need to be downloaded here, just click on the DOWNLOAD button and the download begins. If a download password is set by the PLC, a DOWNLOAD PASSWORD VERIFICATION dialog box will pop up, which is illustrated as follow:

Enter the correct password and click on the OK button. If the PLC is in operation, the following dialog box will pop up to ask the user that if the PLC need to be stopped before the download begins:

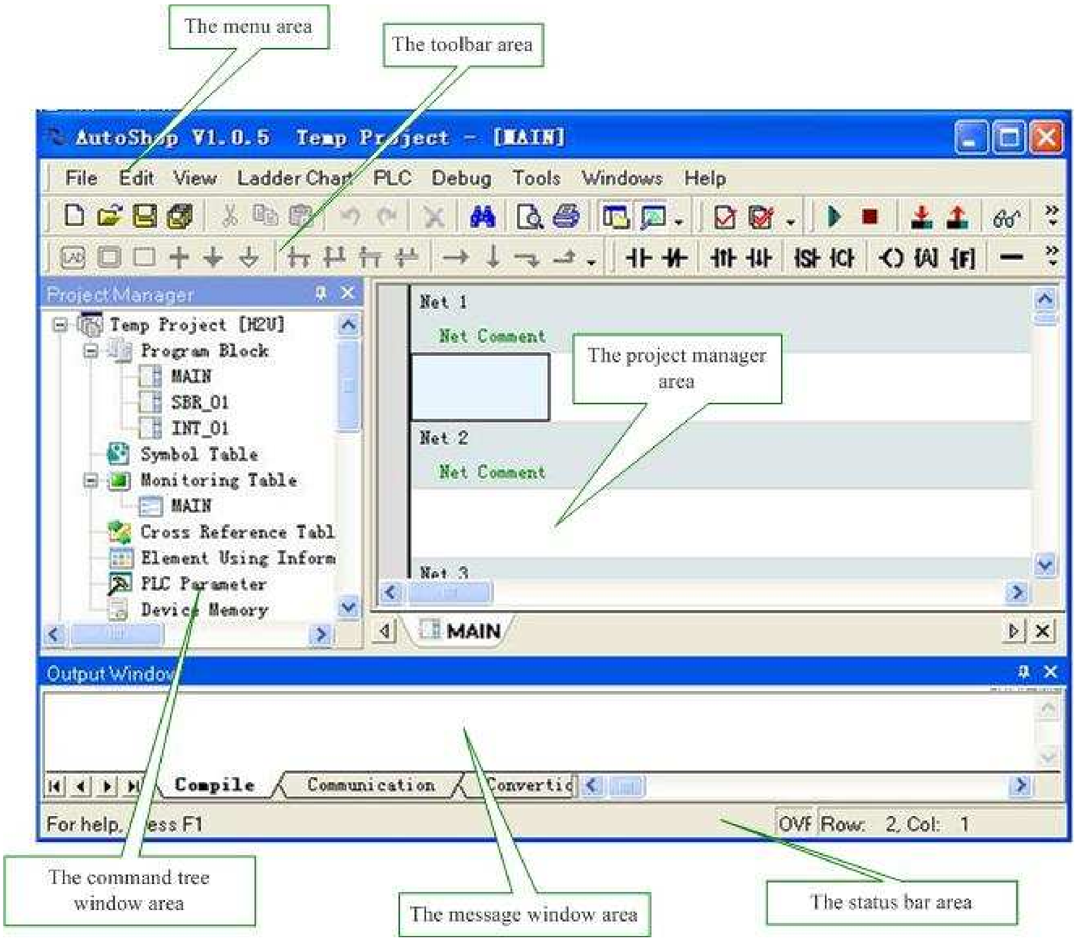
Click on OK and a download progress bar appears, as shown below:

Start programmable controller

After the program download is finished, the PLC should be initiated. Click on "PLC" --- "Start" or toggle the switch of PLC to RUN to start the PLC. If the PLC operates normally then closing the switch of input point X20 will allow for the three lamps connected to output port Y0, Y1 and Y2 to illuminate for the given preset time.

General

The main interface of Autoshop contains seven sections: the menu area, the toolbar area, the project manager area, the command tree window area, the message window area, the status bar area and the working area.



Menu

A menu is consisted of a set of submenus that contain the complete commands.

When the mouse is hovering on a menu item, a brief description about the function of this menu item will be displayed in the status bar. All the submenus will be further explained about the application in the subsequent sections. Submenus are listed as below (note: the items of these submenus might vary slightly depending on the program functions in use).

FILE	The FILE submenu can be used for NEW PROJECT, OPEN PROJECT, CLOSE PROJECT, CLOSE FILE, SAVE FILE, SAVE PROJECT and FILE SAVE AS. It also contains commands relating to print, print settings and print preview.
EDIT	The EDIT submenu contains all the commands necessary for editing, such as undo/redo, insert, cut and paste. In addition, it provides functions of searching of text and graphs, as well as replacing of text strings.
VIEW	The VIEW submenu provides functions to display/hide various windows and toolbars, and to view the current program in different languages.
PLC	The PLC submenu provides functions of interaction with PLC hardware and compilation of the program.
TOOL	The TOOL submenu provides functions to configure the relevant properties and project settings.
WINDOW	The WINDOW submenu provides functions to visit the currently opened windows, and to rearrange the current windows as required.
HELP	The HELP submenu contains a help system about how to use Autoshop.

Besides of the basic menus mentioned above, Autoshop will also display the corresponding menus depending on the window type and program type.

Toolbar

The program provides several toolbars, including various command buttons for quick access to frequently used operations. These operations can also be carried out using the menu items or the pre-set shortcuts.

Toolbars appear above the menu bar and are visible by default. To hide/display a toolbar, right-click on it and select/cancel it in the pop-up shortcut.

While hovering the cursor over any icon (not clicking it), a short description text, called a Tool-Tip will appear,. The Tool-Tip displays the name of the icon. If a Tool-Tip does not appear, enable it using the OPTION Dialog Box.

1.Standard Toolbar:

The standard toolbar contains the basic functions most commonly used to edit a PLC program.

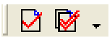
For example:

NEW PROJECT, OPEN PROJECT, SAVE FILE, SAVE ALL, CUT, COPY, PASTE, UNDO/REDO, DELETE, FIND, PRINT PREVIEW, PRINT, DISPLAY/HIDE PROJECT MANAGER, DISPLAY/HIDE INFORMATION OUTPUT WINDOW.

2.Tab toolbar

The tab toolbar is mainly used with the instruction list text editor, to provide tabs in the text window for quick positioning.

3.Compilation Toolbar:



The compilation toolbar contains the functions most commonly used for the two types of compilations: 1) compile current program and 2) compile all programs. That is, to compile the user program in the current window or all the user programs.

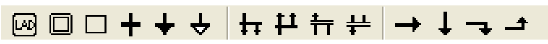
4.PLC Toolbar:

The PLC toolbar provides the functions most commonly used to operate and access the PLC hardware, including the start/stop control of the PLC, the upload/download of the programs, monitoring, etc.

5.Ladder-Chart Toolbar:

The ladder-chart toolbar contains the functions most commonly used to edit ladder-charts.

6.Sequential Function-Chart Toolbar:



The sequential function-chart toolbar contains the functions most commonly used to edit sequential function-charts.

7.Scale Toolbar

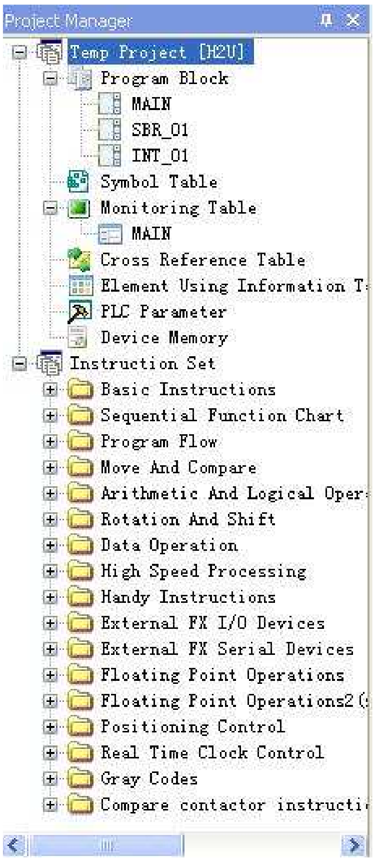
In ladder-chart programs and sequential function-chart programs, the scale display can be adjusted via the scale toolbar.

Addition:

The interface menu contains the menu items corresponding to the functions of each toolbar. It is possible to create custom toolbars or to add new toolbars to display commonly used icons.

Project management and command tree window

As the name suggests, the project management and command tree window are divided into two modules: the project management module and the command tree module.



1. The project management module can be subdivided into 6 parts: program blocks, symbol table, component monitoring table, cross reference table, system parameters and soft component memory. The following functions can be achieved via the command tree:

1. Right click on the project name node, and select SAVE PROJECT, PROJECT SAVE AS, CLOSE PROJECT or MODIFY PROJECT PROPERTIES etc. in the pop-up menu;

2. Right click on the program block node, and select INSERT SUBROUTINE or INTERUPT SUBROUTINE;

3. Right click on one of the program nodes under the program block, and select OPEN PROGRAM, DELETE PROGRAM (the main program can¡¯t be deleted) or MODIFY PROPERTIES in the pop-up menu;

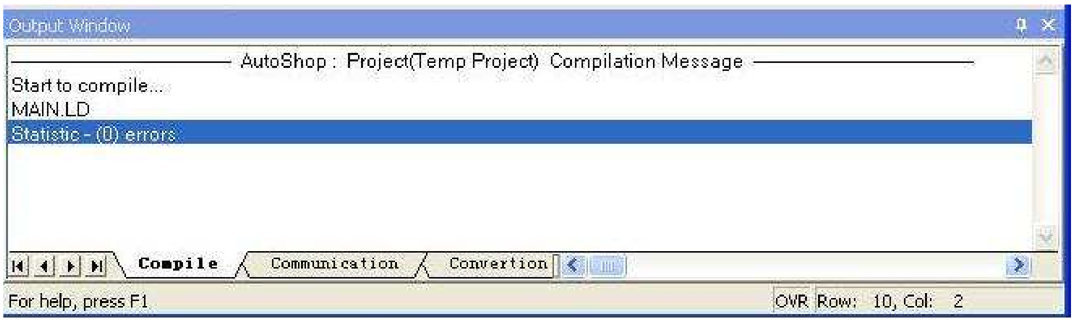
4. When right clicking on other nodes, only the Open operation is accessible in the pop-up menus;

2. The command tree module includes all the commands supported by the SFC, the ladder chart, and the command table. When programing codes in the editing window use different languages, contents of the command tree will change according to the change in the editor in the current working area window. If the current working area is the ladder editor, the command tree will display all the commands supported by the ladder chart language. Likewise, when editing program uses the command table or the SFC, the command tree will display the corresponding command scope. The command tree can be used through the following two ways:

1. Under program editing mode, double click on a node of the command tree and a command help window will pop up, through which the user can generate the corresponding commands.

2. Select a node of the command tree using the left button of the mouse, and then hold the left button and drag the node into the code editing area. If the right position has been dragged to, a command help window will pop up, through which the user can generate the corresponding commands.

Information output window



The information output window can provide the user with the results after Autoshop has executed the operations, including the result information of the three types of operation: compilation, communication and conversion.

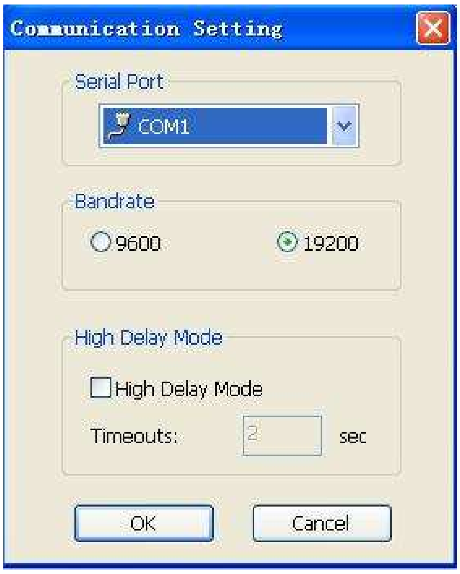
Working space

The working space contains:
the program editing dialogue box,the global variable table editing dialogue box ,and the cross-reference table dialogue box.

Communication configuration and system option

There are two items in the TOOL menu: communication configuration and systems option.

The communication configuration diagram is shown below:



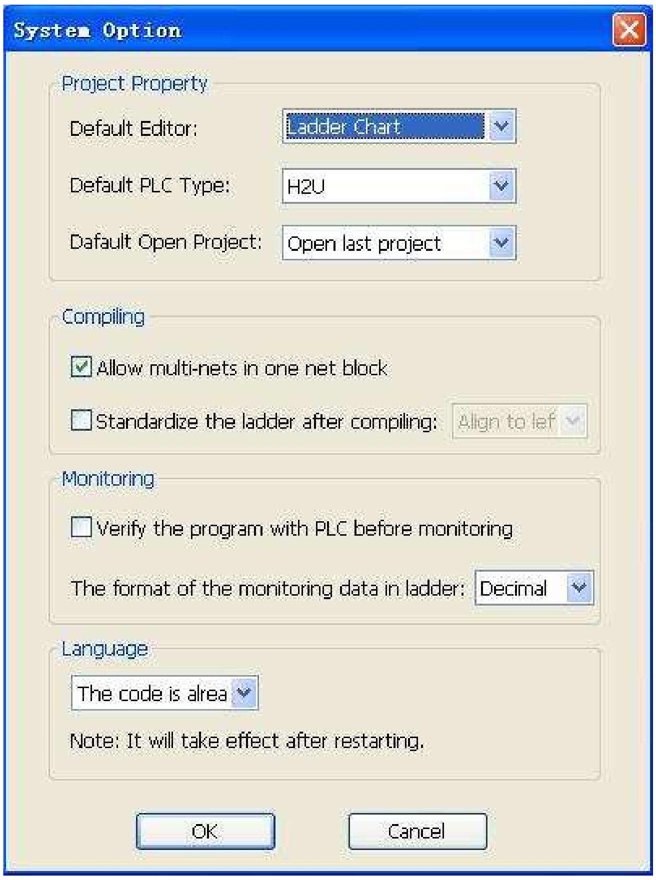
The settings of communication configuration contain three aspects:

1) Setting the serial connection between the PLC and the PC.

2) Setting the communication rate between the PLC and the PC, namely the data that will be transferred per second.

3) Setting high delay mode.

The systems option diagram is shown below:



The systems setting contains the following aspects:

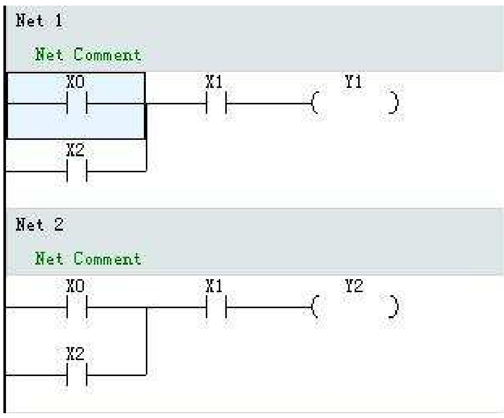
1) Default editor: That is the editing environment of the PLC programs (one of the ladder chart, the SFC and the instruction table).

2) Default PLC type:That is one of the PLC types of INOVANCE when project created(one of the H1U,H2U,MDI card).

3) Default open type: That is which project auto opened when Autoshop first run(Open last project,crecte temp project,do nothing).

4) Compilation: If "multiple networks in one network block" is selected, more than one output is available in one network block; inversely, only one output is available in one network block.

For example: if the option of "multiple networks in one network block" is selected, programs can be edited in such a way for one network block without an error after compilation.



5) If "Standardize the ladder after compiling" is selected,standardize the ladder after compiling video mode can be set(Align to left,Align to right).

6) Monitoring: if "Verify the program with PLC before monitoring" is selected,when pash Monitor button,project will be checked

fristly.

7) If "The format of the monitoring data in ladder" is selected,Decimal and Hexadecimal can be choosed.

8) Language:sofeware support Simplified Chinese and English .Note:It will take effect after restarting.

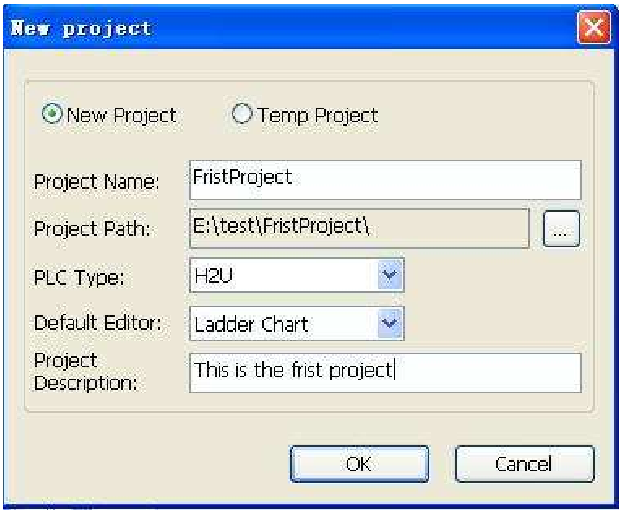
Using of online help

The Autoshop provides extensive online help. When a problem occurs during use of the software, you should first refer to online help. You can get online help by selecting HELP TOPIC in the HELP menu from any dialogue box in the software.

Project usage description

Various types of files are used during the program development process with Autoshop, including program blocks, global variable tables, component monitoring tables, memory blocks of soft components, systems blocks, component cross reference tables, etc. All of these various types of files are managed in Autoshop using projects.

The extension for project files is hcp. Projects have the following properties which can be selected or entered in the NEW PROJECT dialog box:



PROJECT NAME: Identifier for a project as well as the file name when saving the project to disk.

PATH NAME: The fully specified path under which the project file is saved. The final directory for the project is the subdirectory with the project name (created automatically) added to the specified path.

PLC TYPE: Indicates the PLC type the project is oriented to. Selecting different PLC types/versions will affect the following: the CPU instruction set and the program volume supported by the system block configuration. The program volume will affect the compilation and download progress. Therefore, volume inspection is required when compiling programs. If there are more program steps than the set value, a warning message should be displayed and an error message should be displayed while downloading.

EDITOR TYPE: Three editor types are available: the ladder chart, the instruction list, and the sequential function chart.

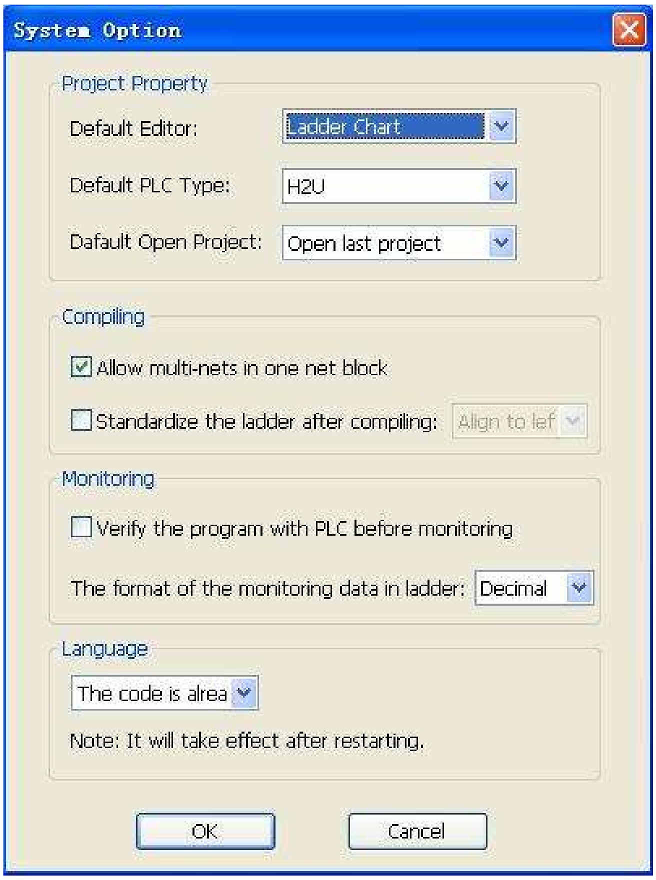
PROJECT DESCRIPTION: A simple description of the project. May be blank or up to 128 chars.

When downloading a program to the PLC, the software will compare the PLC TYPE defined in the project and the actually connected PLC type. Downloads will not be able to continue if the types don't match.

To change the PLC type of a project, select the CHANGE PLC TYPE item in the FILE menu, or select the root node of the project tree and select PROPERTIES in the right click menu. In the pop-up project properties dialog box, all the properties are modifiable including PLC TYPE, except for project name and save directory.

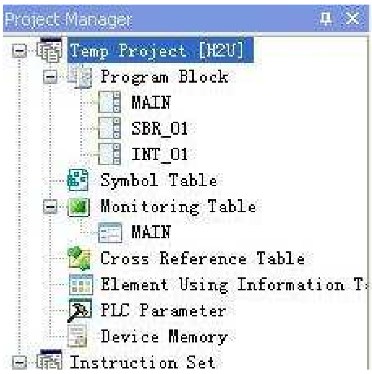
Note:
Changing the PLC type might cause the project to fail the compilation. This is possibly because of incompatible commands or system block configuration. Manually modify the settings according to the compilation prompt questions.

You can specify the default PLC type and editor type for new projects in the system options dialog box, as shown below:



Each project typically has seven files: Config.sdt, CrossTable.crs, VarList.gdt, MAIN.mon, MAIN.LD, INT_01.LD and SBR_01.LD, which are used to store the system parameter table, the cross reference table, the symbol table, the soft component monitoring table, the main program, the subroutines and the interruption subroutines respectively.

The project management of Autoshop is performed in the window as illustrated (described in the section of project window and the command tree window):



Note:
Do not directly delete or modify the files mentioned above, or the project might not be able to start up normally, and the program data might be damaged.

General

The software provides three different types of program units: the main program, subroutines and interruption subprograms.

Main program: The software provides only one main program by default. This is the program that the PLC executes at the RUN command, and it can be written in any form of ladder chart, sequential function chart or instruction list language.

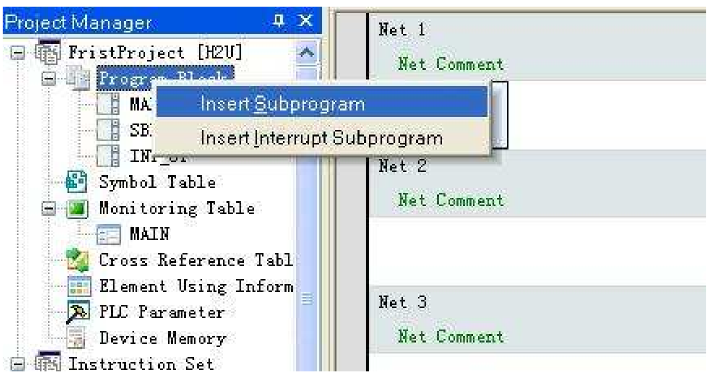
Subprogram: A project may contain multiple subprograms, but it cannot have more than 127. Subprograms may be called up by the main program or other subprograms to perform some common functions, or functions that are used repeatedly. Subprograms can only be written using ladder charts or instruction lists, rather than sequential function charts.

Interruption subprogram: A project may contain interruption subprograms, but it cannot have more than 21. Interruption subprograms may be called up by the main program or other subprograms to perform some common functions, or functions that are used repeatedly. Subprograms can only be written using ladder charts or instruction lists, rather than sequential function charts.

The purpose of using subroutines is to partition and divide the program, and they may be written for commonly used function modules that can be executed repeatedly afterwards. The PLC can be used more efficiently when it uses smaller program blocks that it only executes when necessary. Because not every program block might need to be scanned each time the main program calls up a subprogram to run, the subprogram executes all its instructions until the end. Then the system returns control back to the main program that called up the subprogram.

Create subprogram

Right click "Project management" window program block. Select the insert subprogram or insert interrupt subprogram. The default name of the new subprogram name is SBR_ *,and the default name of the new interrupt subprogram name is INT_ * (where * is a number automatically calculated by the software). After the establishment, you can modify the default program name into a more meaningful name through the subprogram properties dialog box. Operation of the new subprogram as shown below:



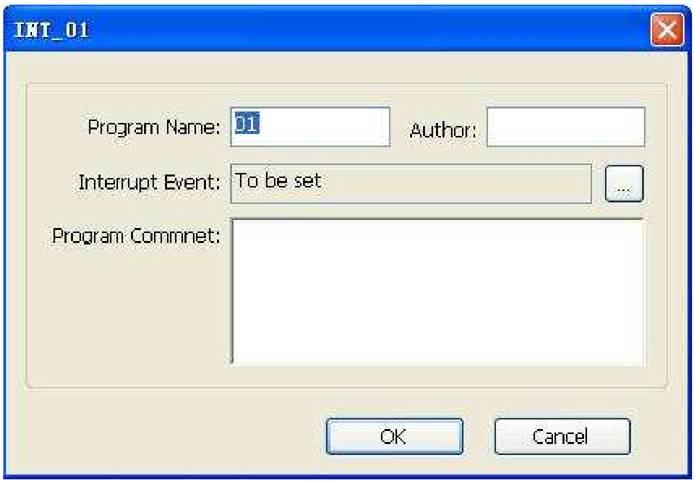
The program will also be opened when the project tree is inserted new program node, and you can edit it immediately.

Modify the subprogram (interrupt subprogram) property

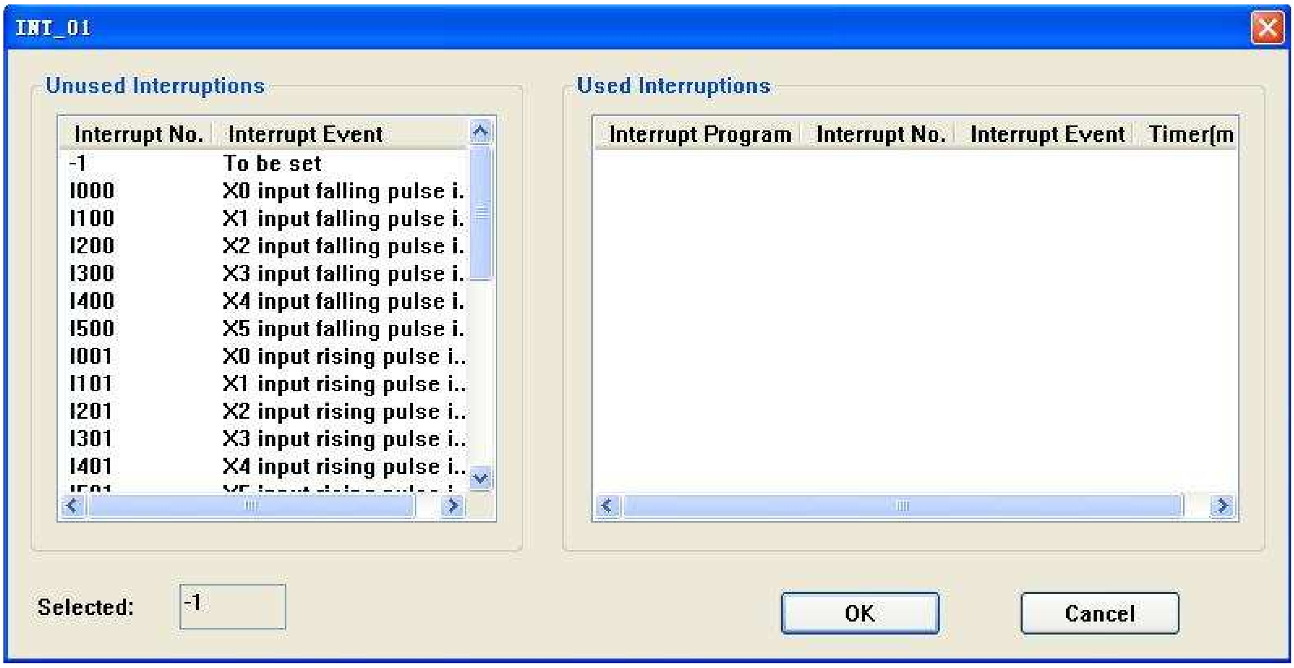
Select the subprogram node that need to be modified in the project tree. Select [Properties (P)] from the right-click menu. Open the file-properties dialog box. You can modify the program name, author and program description of the subprogram in the subprogram properties dialog box, as shown below:

For the interrupt subprogram, you can assign interrupt number (the default interrupt number of the new interrupt subprogram is

-1. That is not set). The interrupt subprogram properties dialog box as follows:



Click "..." button and then pop up interrupt distribution dialog. You can see currently available idle interrupts and other interrupts that have been used by other interrupt subprogram. You can choose an interrupt number in the idle interrupts and assign the number to the current interrupt subprogram, as shown bellow:



As some interrupt events of the system is conflicting. For example, X0 rising edge and X0 falling edge can not be answered by the system at the same time. Thus, if the X0 rising edge interrupt is assigned, X0 falling edge interrupt can be no longer used. In this case, X0 falling edge interrupt will be blocked in the idle interrupt. Conflicting interrupt events as follows:

X0 rising input edge interrupt (interrupt number: 0) X0 falling input edge interrupt (interrupt number: 1)

X1 rising input edge interrupt (interrupt number: 100) X1 falling input edge interrupt (interrupt number: 101)

X2 rising input edge interrupt (interrupt number: 200) X2 falling input edge interrupt (interrupt number: 201)

X3 rising input edge interrupt (interrupt number: 300) X3 falling input edge interrupt (interrupt No.: 301)

X4 rising input edge interrupt (interrupt number: 400) X4 falling input edge interrupt (interrupt number: 401)

X5 rising input edge interrupt (interrupt number: 500) X5 falling input edge interrupt (interrupt number: 501)

Note:

Modified subprogram name is always prefixed by SBR_; Modified interrupt subprogram name is always prefixed by INT_ .

Some rules for using the subprogram

1.In the main program, you can nest subprogram (place subprogram call instructions in the subprogram). The maximum nesting depth is 5. Call subprogram is not allowed in the interrupt program.

2.Cycle call is prohibited between user programs. For example, subprogram A and B call each other.

3.User program prohibits recursive call. For example, user program A calls program B, program B calls program C, then program C calls the procedure A. Thus they form a ring. Besides, subprogram is not allowed to call the subprogram itself.

Description of data types

The system supports four data types: BOOL, 16-bit integer, 32-bit integer and FLOAT, each is detailed in the following table:

Data type	Description	Bits	Range
BOOL	BOOL	1	ON¡¢OFF
16bit integer	word	16	0~65535
32bit integer	double word	32	0~4294967295
FLOAT	float	32	±1.175494351 E - 38 ~ ±3.402823466 E + 38

Symbol table definition

Table heading includes three attributes which are sybol name, address and comment, and the address in fact is equivalent to component name.

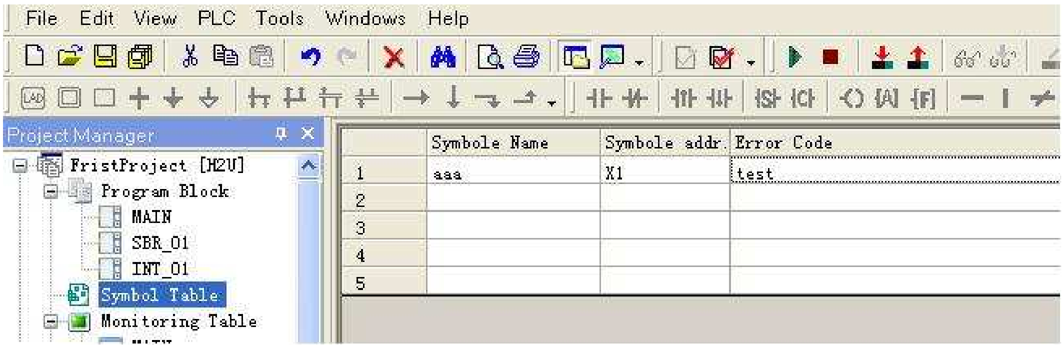
The rule of symbole definition: Symbols must be composed of A~Z, a~z, 0~9, underline and Chiniese character, and symbol name can't start with number or be only numbers. The symbol name is not case sensitive, and the length can't exceed 16 english characters. "Component type character + numbers" can't be used as program and variable name. The symbol name can't contain space or reserved keywords such as fundamental data type name, instruction name and operators of instruction list.

Use of the symbol table

The symbol table is mainly used to modify addresses (soft component symbols). Symbol names can be used to replace addresses during the programming process. A symbol name is much like an alias of a certain address, which makes it simpler for the program to understand and identify. Annotations are descriptions on the meaning of addresses. Those are to help users to gain a better understanding about the program.

Edit symbol table

Double click on the symbol table in the project management window to open it.



Symbol:

A symbol represents an address that can replace the address when programming.

Address:

That is the name of a soft component.

Annotation:

That is used to give information about the address and is limited to 20 chars (10 Chinese characters). Annotations can be added in symbol table or in ladder editor, for details refer to adding network annotations and component annotations in ladder charts .

Symbol table provides the following editing functions:

Ordering:

Click on the head of any column of the symbol table using the mouse to enable the ordering of that column of symbols, component addresses or component annotations in ascending or descending.

Find, replace and locate:

Find certain words in the symbol table, the target found will be indicated by the cursor in the cell containing it. Click on the "FIND NEXT" button to move on and find the next cell that satisfies the conditions. The words found can be replaced by words otherwise specified. It is up to the user to decide if only the current words found will be replaced or all the matched words will be replaced automatically.

Cut, copy, paste and delete:

The words in one or several cells can be copied or cut, and then pasted to multiple cells selected in the same column. During the cutting, copying and pasting of symbol names and component addresses, the system will not pop up prompt window for identic contents of multiple cells, which will only be indicated in red.

Insert a line:

To insert a new line above the selected line

Delete a line:

To delete a new line above the selected line

Add a line:

To add a new line at the end of the cells

Undo and redo:

To undo an incorrect operation during the edit process, you can either click on the UNDO button in the commonly used toolbar, or you can click on the UNDO item in the EDIT menu. A maximum of up to 20 latest operations can be undone. Also, you can click on the UNDO button in the commonly used toolbar or on the UNDO item in the EDIT menu to redo the last operation you have undone. A maximum of up to 20 latest operations can be redone.

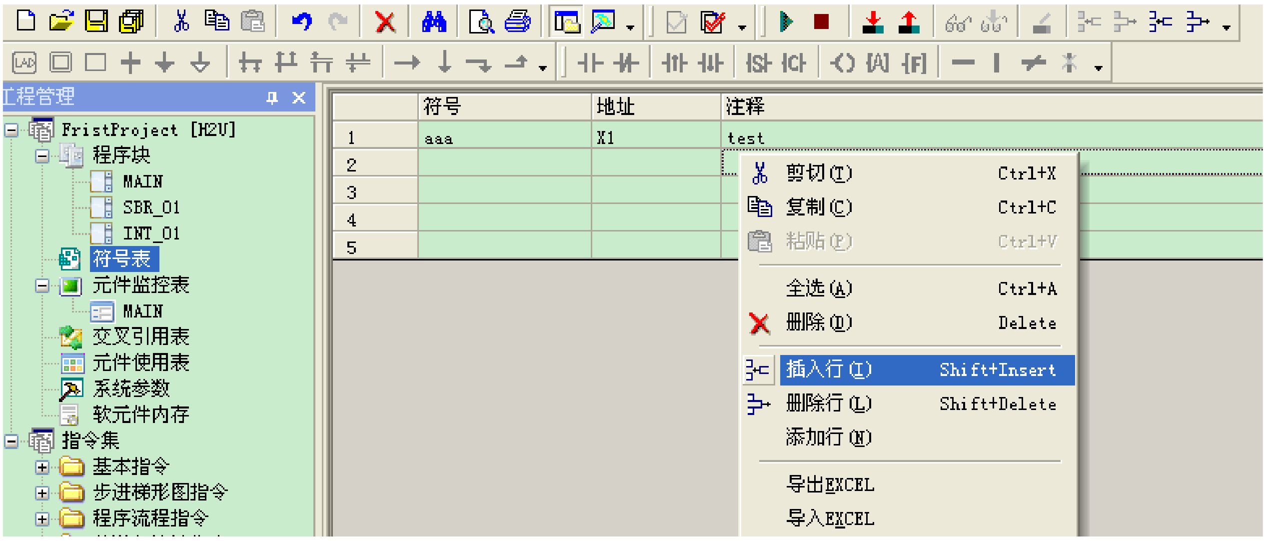
Export EXCEL:

Export the symbol table to EXCEL，you can edit it in EXCEL before import to symbol table，modified usefully .

Import EXCEL:

Import the excel to symbol table which is edited in EXCEL.The order of get item accrroding to symbol table's item in col 1,in row 1 in correspondence with item in col 2,in row 2 in EXCEL.

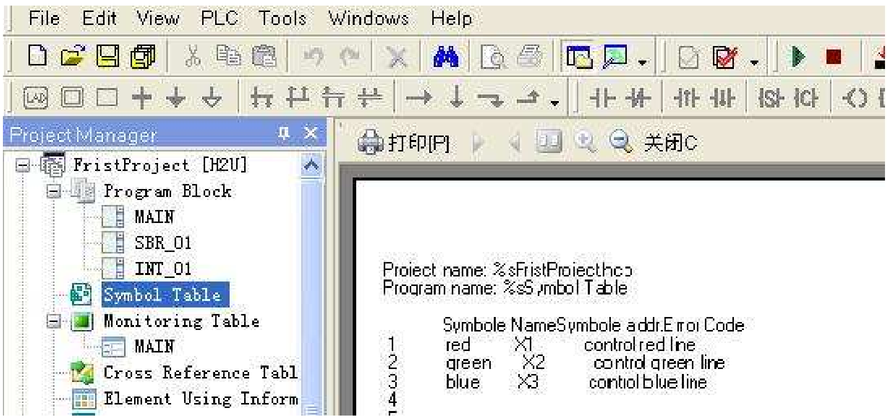
You can either select CUT, COPY, PASTE, DELETE, INSERT A LINE, DELETE A LINE or ADD A LINE in the EDIT menu, or you can right click on the symbol table and select these items in the pop-up menu, as illustrated in the following figure:



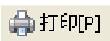
Printing the symbol table

Print preview:

You can preview the printing on the display. To do this, click on the [FILE/PRINT PREVIEW] command or the  button in the toolbar, as illustrated below:



The preview toolbar in the preview window has following functions:



: Print the current program.

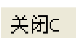
: View the next page.

: View the last page.

: Double-paged display.

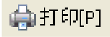
: Scale up.

: Scale down.



: Close the preview window.

Print:

Click on the() button in the toolbar or on the (

) button in the print preview window to print. At first a standard print dialog box will pop up. You can configure print options at this point. After print confirmation the current program will be printed.

Definition of a component monitoring table

The header of a component monitoring table comprises COMPONENT NAME, DATA TYPE, DISPLAY FORMAT, CURRENT VALUE and NEW VALUE, which are explained respectively as follows:

COMPONENT NAME:

The name of the soft component.

DATA TYPE:

The data type of the soft component; refer tothe description of data types

DISPLAY FORMAT:

Shows the value of the soft component, which may be displayed as binary, decimal or hexadecimal.

CURRENT VALUE:

The currently displayed value of the soft component during monitoring mode.

NEW VALUE:

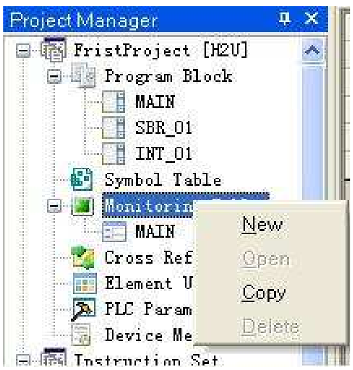
The value of the soft component can be caused to change by entering a new value in the NEW VALUE column. Besides enforcing a change of the value of the soft component in the monitoring table during monitoring mode, a change may also be made in the ladder editor. For details refer to enforcing change of the soft component value in the ladder diagram during monitoring mode

The Role of the Monitoring Table

The monitoring table is mainly used to supervise the value of software components in monitoring mode in real time, which can be helpful for debugging.

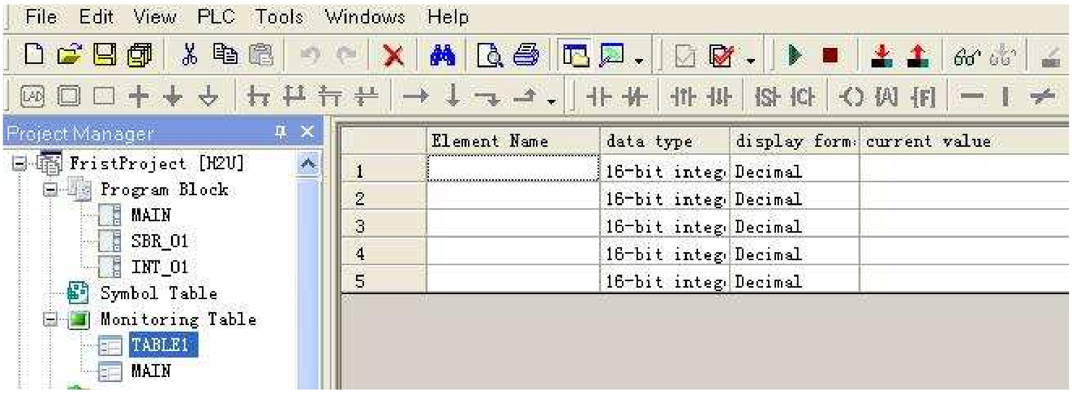
Build New Component-Monitoring Table

When building new a project, the system will by default generate a component-monitoring table named "MAIN" in the project management window. Also, if you want to create more than one component monitoring table, you can right-click "component-monitoring table" in the pop-up menu, select "New" menu item, and then you can create a new control table. As shown below:



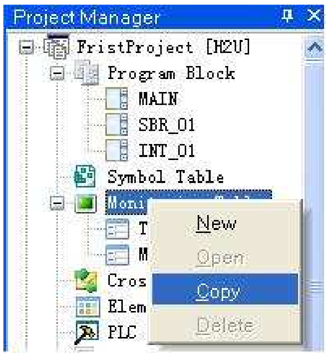
After selecting "New", a dialog box for creating a component-monitoring table will pop up. From here, you can enter the name "TABLE1" in the editing box, as shown below:

Click "OK", then the monitor table "TABLE1" is created successfully, as shown below:

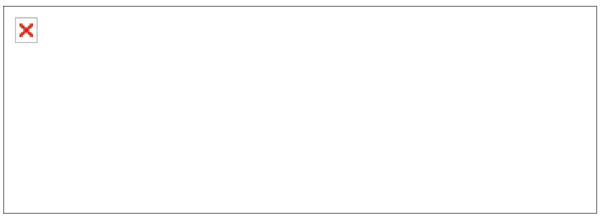


Copy Component Monitoring Table

Right-click component-monitoring table in the project management window and select "copy" in the pop-up menu:

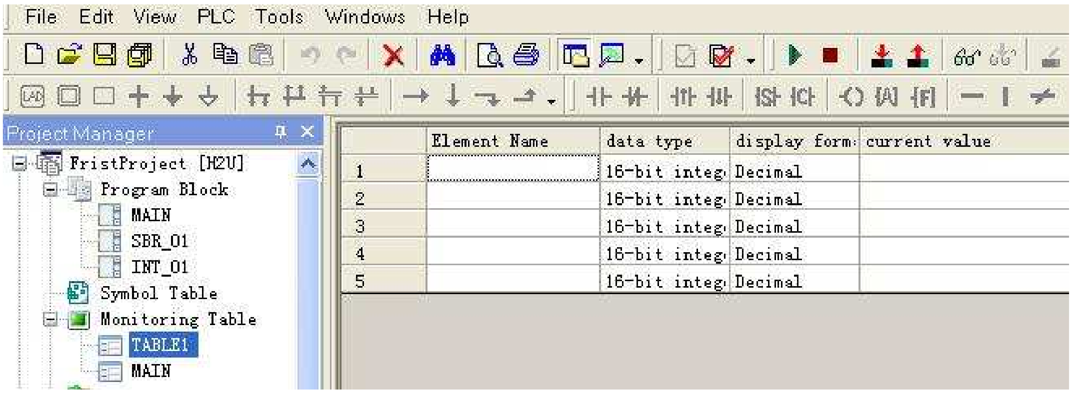


From here, a dialog box for the duplication monitoring table will pop up, as shown below:



First, select the name of monitor table that needs to be copied, then enter the new name of the monitoring table, click "OK," and a new table will be created, as shown below:

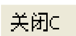
Click "OK", then the monitor table "TABLE1" is created successfully, as shown below:



k:@MSITStore:C:\Inovance Control\AutoShop\AutoShop.chm::/Quick_start/Symbol_table/../images/Double-paged%20display.bmp" />: Double-paged display.

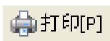
: Scale up.

: Scale down.



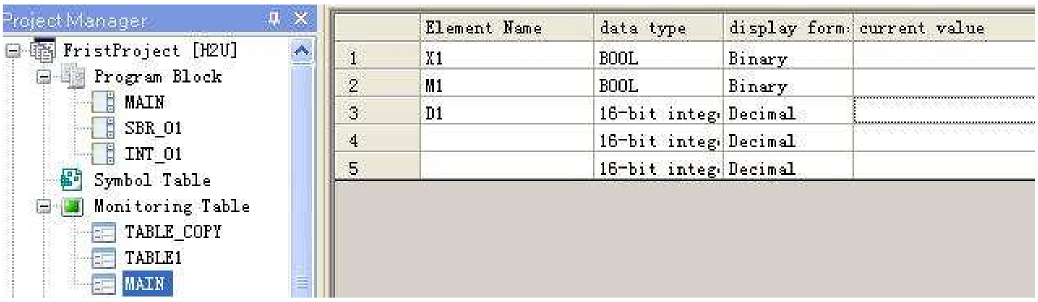
: Close the preview window.

Print:

Click on the() button in the toolbar or on the () button in the print preview window to print. At first a standard print dialog box will pop up. You can configure print options at this point. After print confirmation the current program will be printed.

Edit Component-Monitoring Table

The component-monitoring table can be opened in the project management window by double clicking, as shown in this example:



In monitoring mode, when the value of a component needs monitoring, it is entered in the NAME of the given component, so that the system can judge the type of component data according to the component and display the current value automatically.

The editing function provided by component-monitoring table is similar to the symbol table:

You can check the symbol table for more detail

Print Component Monitoring Table

The printing function provided by component-monitoring table is similar to that of a symbol table. You can check the related function in symbol table.

Cross reference table definition

Users can view the cross reference table which shows the program component usage. Double click on the record item in the cross reference table or select "component position" from the right click popup menu to find the program and position (row and column) in which this component exists.

The cross reference table can't be edited, only looked-up. Clicking the "edit"---"lookup" menu item or () button activates the lookup function. Double clicking on any cell in the cross reference table will take you to the component location in the program editor.

The cross reference table will automatically refresh program information when a program unit is saved. The information included in the cross reference table contains following items:

Element: If the component is defined as a variable in the global variable table and the current view mode is set to the variable mode, then the component variable name or the component address will be shown.

Block: User program in which component exists.

Location: Row and column number of the component in the user program.

Context: The manner of component usage

Printing cross-reference table

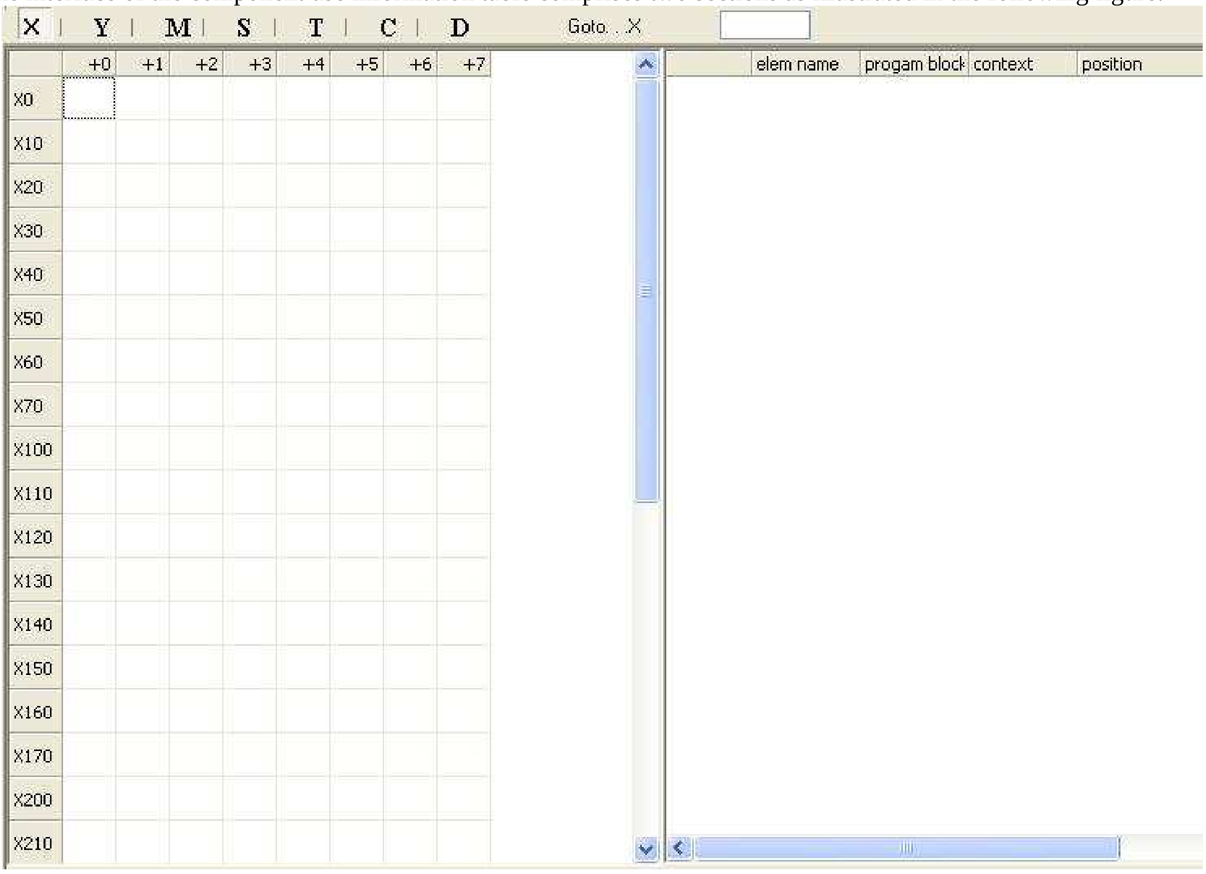
The function to print the cross-reference table is similar to that of symbol-table. Please refer to the symbol-table print function

Summary of the component use information table

The component use information table is mainly employed by the user to view the situation in which a certain type of component is used, and to display the information from the corresponding cross-reference table.

If the program has been edited by the user, it has to be recompiled before the content in the component use information table can be updated.

The interface of the component use information table comprises two sections as illustrated in the following figure:



It can be seen from the figure that the soft components used by the program are checked (using "√") to the left of the component use information table. You can click on the component button above the table to switch to the corresponding component use information. If you want to quickly locate a soft component, you simply enter the component address in the edit box above the table and press the ENTER button.To the right you may view the cross-reference table for this type of soft component.

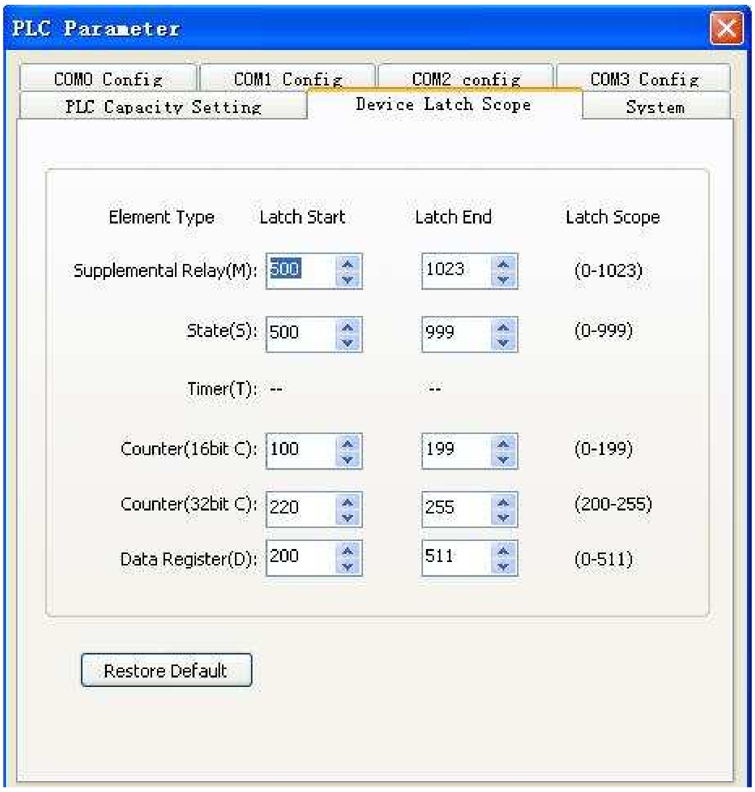
Summary of the system parameters

Open the system parameters window by double clicking on the button named "system parameters" in the project management window. The parameters of each part are explained below as follows:

Memory volume setting:

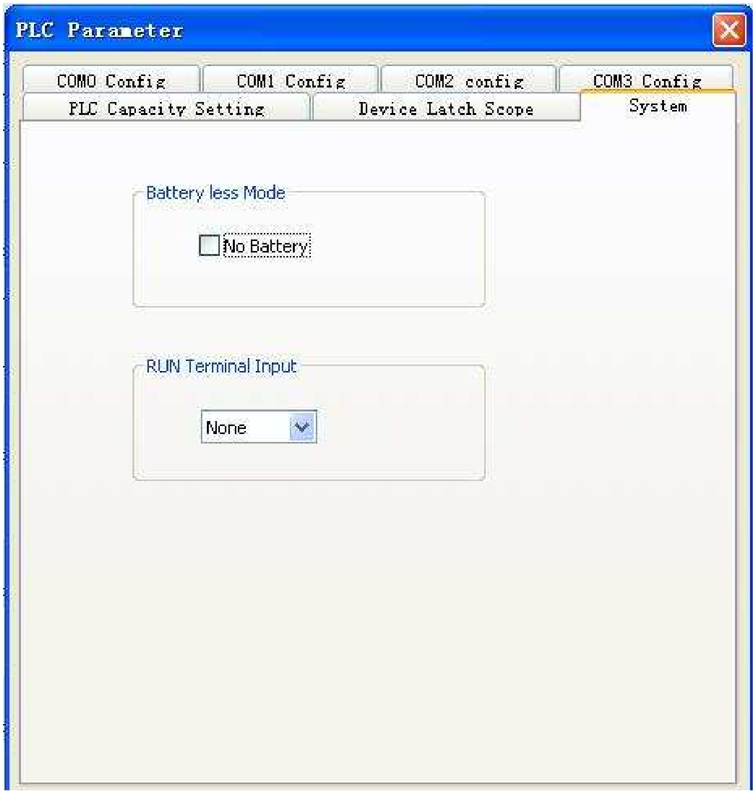
1.Memory volume:

The capacity of the system, including the sum of the annotation volume, file register volume and program volume. As shown in the above figure, the memory volume is configured to 16000 steps, with a size of 32000 bytes.



Range of power-off protection:

It is indicated in the above figure that: for the values in the time setting ranges of an auxiliary relay (M), state relay (S), timer (T), counter (C) and data register (D) components, they will be automatically saved to the Autoshop system files when a power-off occurs to the PLC. Thus, the user can prevent the data in the PLC being damaged under abnormal conditions.



System:

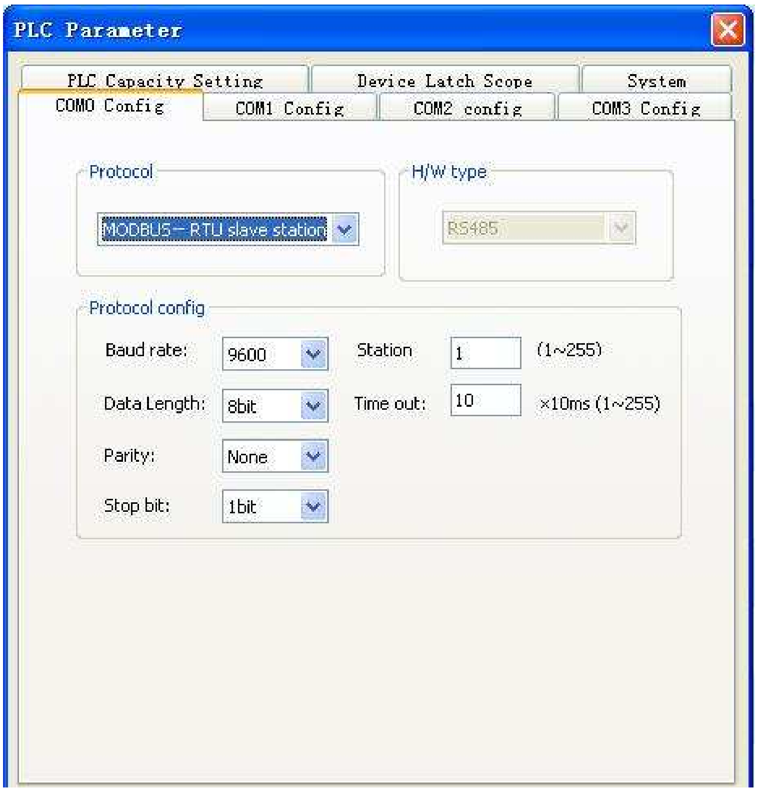
1.No battery mode:

When setting the operation mode of the PLC, toggle the radio button to activate the no battery mode, and vice versa means the battery mode is active.

2.Operation terminal inputs:

The user can select one input terminal from X0~X17 of the PLC to control it. It is also possible that the internal program of the PLC will automatically do the job without interference from the user.

Com0 config:



1)Protocol:

Three protocol can selected in combo box which are "Download/HMI monitor protocol","MODBUS-RTU slave station","MODBUS-ASC slave station".

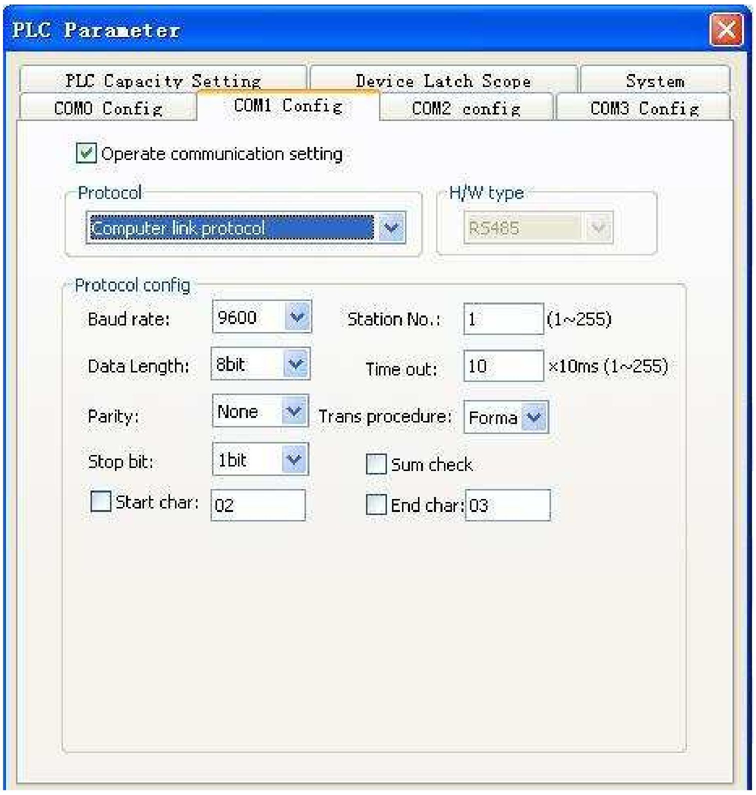
2)H/W type

When selected "Download/HMI monitor protocol" ,RS232c/RS422 and RS485 can be selected.If the user selected "MODBUS-RTU slave station" and "MODBUS-ASC slave station" H/W type default is RS485.

3)Protocol config:

The user can set the config about protocol.

Com1 config:



1)Protocol:

Nine protocol can selected in combo box which are "Non-procedural","HMI monitor protocol","MODBUS-RTU host station protocol ","MODBUS-ASC slave station protocol","Computer link protocol","Parallel connection protocol host station","Parallel connection protocol slave station","N:N protocol host station","N:N protocol slave station".

2)H/W type

When selected "Non-procedural" and "HMI monitor protocol" ,RS232c/RS422 and RS485 can be selected.If the user selected other H/W type default is RS485.

3)Protocol config:

The user can set the config about protocol.

4)Parallel connection protocol

It must be one com be used in this protocol from com1 to com3.

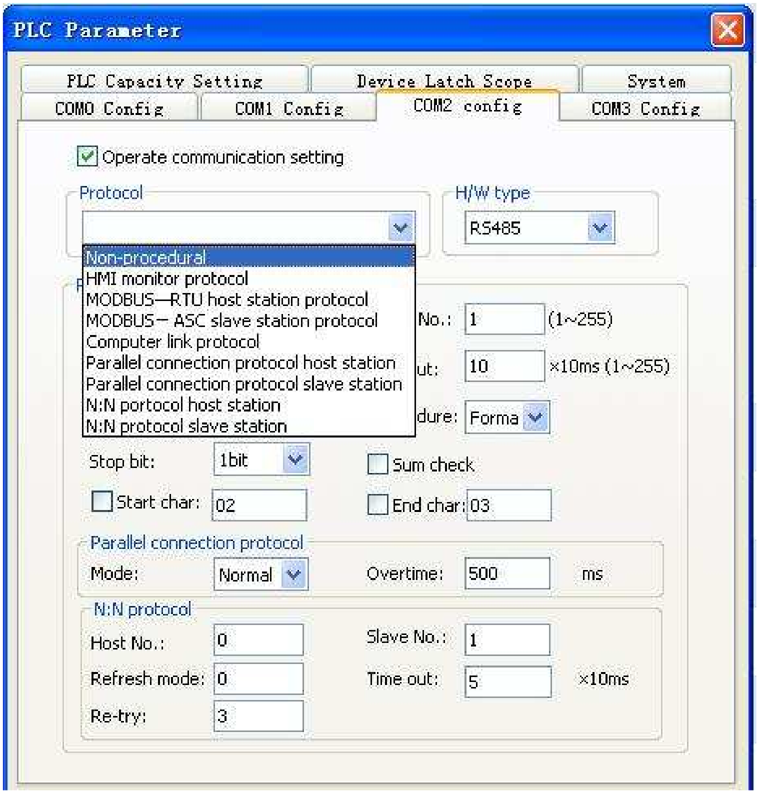
5)N:N protocol

It must be one com be used in this protocol from com1 to com3.

6)Operate communication setting

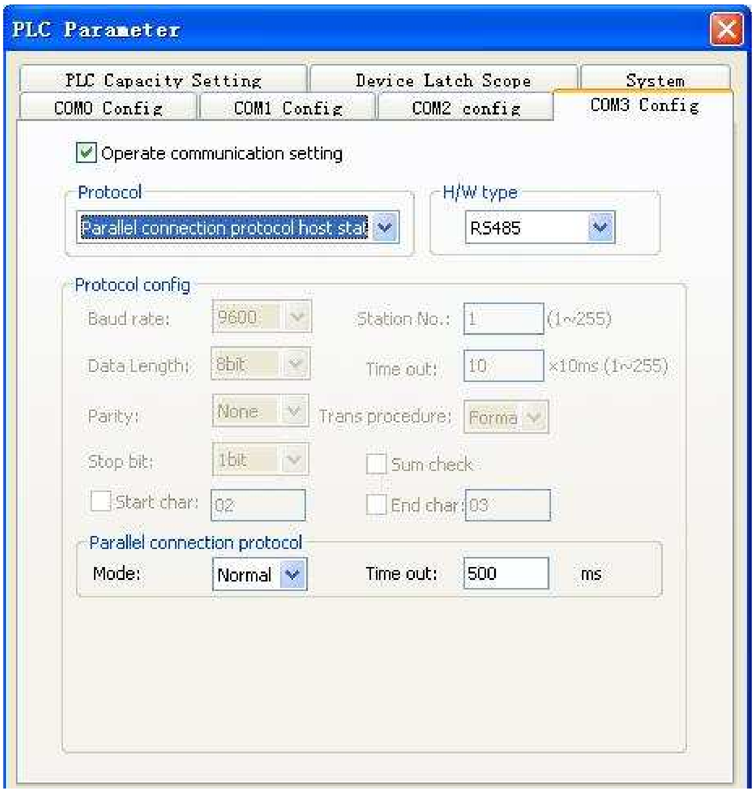
When User check the item,this com is used.

Com2 config:



Similar to that of com1 config.

Com3 config:



Similar to that of com1 config.

Memory table definition for soft elements

The memory table for soft elements is mainly used to protect the values of the data register component (D) set by the user; the values of the component are in the range of D0---D8255.

Each D component has a size of 16 bits, which means that 16 integers can be stored. Therefore, two D components in series are able to store 32 integers, floats or fixed point numbers (under a 32-bit operating system).

Create and copy a soft component memory table

Create a new soft component memory table:

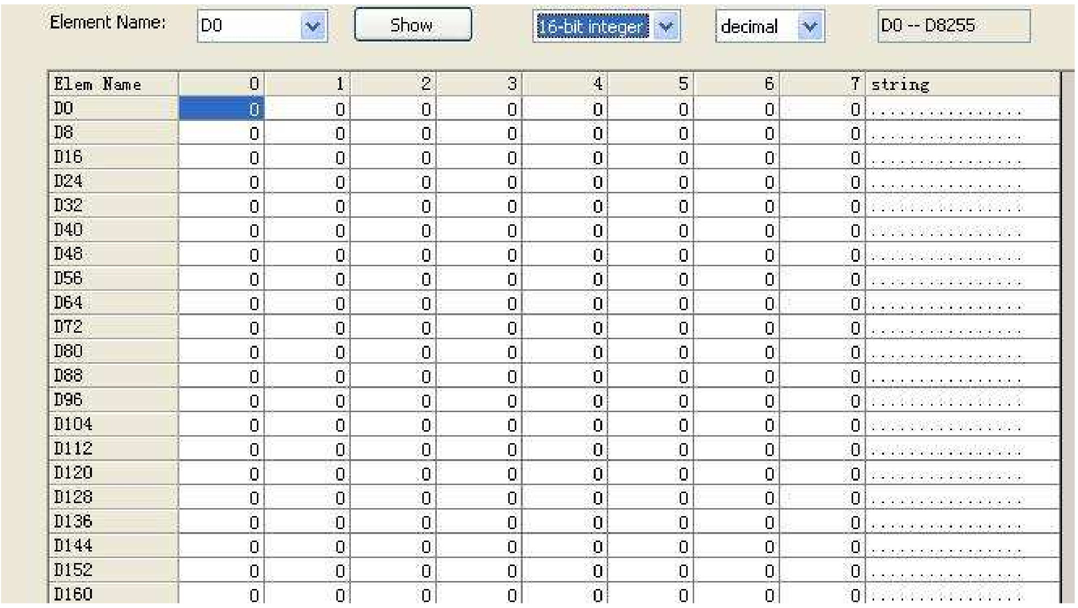
In the Project Management window, right click the soft component memory table. Choose the "New" button in the pop-up menu, and click the "OK" button in the next pop-up dialog box to create a new memory table. The procedures are similar to that of creating a new component monitoring table. Please refer to the operating procedures for creating a new component monitoring table .

Copy the soft component memory table:

In the Project Management window, right click the soft component memory table. Choose the "Copy" button in the pop-up menu, and then choose the name of the soft component memory table to be copied in the pop-up dialog box. Enter a new name for the memory table, and click the "OK" button. The procedures are similar to that of copying a component monitoring table. Please refer to the operating procedures for copying a component monitoring table .

Compiling the soft component memory table

Double-click the file name once you have built a new soft component memory table. You can open the file, as illustrated below:

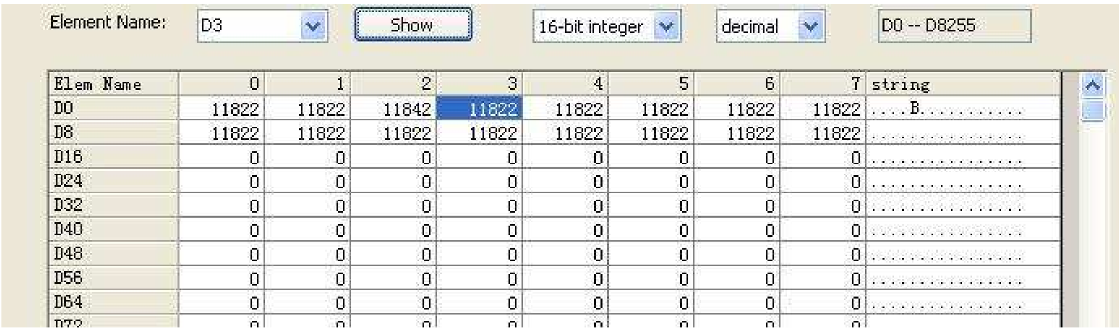


Elem Name	0	1	2	3	4	5	6	7	string
D0	0	0	0	0	0	0	0	0	....
D8	0	0	0	0	0	0	0	0	....
D16	0	0	0	0	0	0	0	0	....
D24	0	0	0	0	0	0	0	0	....
D32	0	0	0	0	0	0	0	0	....
D40	0	0	0	0	0	0	0	0	....
D48	0	0	0	0	0	0	0	0	....
D56	0	0	0	0	0	0	0	0	....
D64	0	0	0	0	0	0	0	0	....
D72	0	0	0	0	0	0	0	0	....
D80	0	0	0	0	0	0	0	0	....
D88	0	0	0	0	0	0	0	0	....
D96	0	0	0	0	0	0	0	0	....
D104	0	0	0	0	0	0	0	0	....
D112	0	0	0	0	0	0	0	0	....
D120	0	0	0	0	0	0	0	0	....
D128	0	0	0	0	0	0	0	0	....
D136	0	0	0	0	0	0	0	0	....
D144	0	0	0	0	0	0	0	0	....
D152	0	0	0	0	0	0	0	0	....
D160	0	0	0	0	0	0	0	0	....

The soft component memory table contain functions such as: locating the D component、the numerical input of the D component,the strings input of the D component,obtaining the input strings of the D component,showing the data types of the numerical value (16-bit,32-bit,floating and fixed point),and the display of the D component value (in decimal).

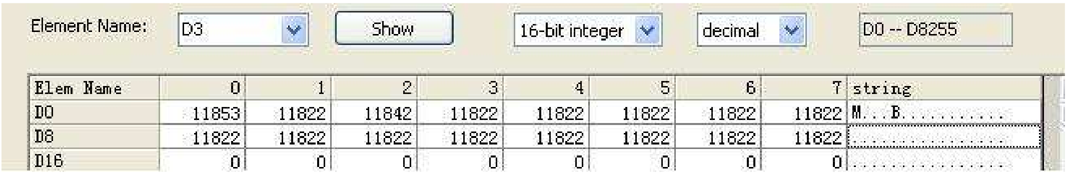
Locating the D component:

As illustrated above, enter the component name in the editing box "the soft component name", press the "Enter", or click the "Show" button. System will navigate to the location of the corresponding D component, and highlight it as below:



The numerical input of the D component:

Take D0 for example. Select the box where D0 lies with the left mouse button, enter a numerical value (in 16-bit) and press Enter. The system will display the corresponding ASCII character of the value of D0 in the last column "string", as illustrated below:

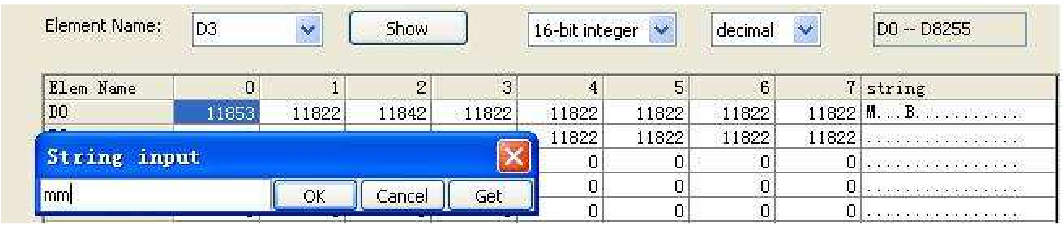


Note:

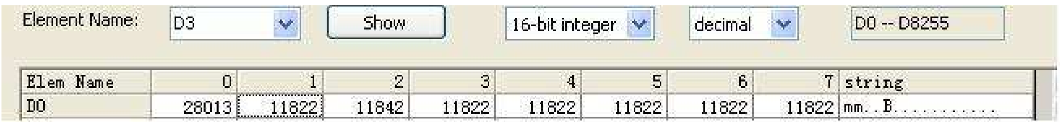
If input is less than 32, the corresponding character in the "strings" column will display with points (take D0 in the last illustration for example, the 16-bit value of D0 is 333. If the value of its low 8-bit value is more than 32, the corresponding ASCII character is M, while its high 8-bit value is 0, which is less than 32, so it will show with points, and the corresponding strings of D0 is "M.").

The strings input of the D component:

Take D0 for example again. When necessary to enter strings for D0, you can double-click the box where D0 lies, then a string input dialog box will pop up as illustrated below:



Click the "Sure" button, as shown in the previous picture. This will enter the "mm" string into the last column and the corresponding 16-bit integer of the "mm" string will be shown in the D0 cell (as shown below):

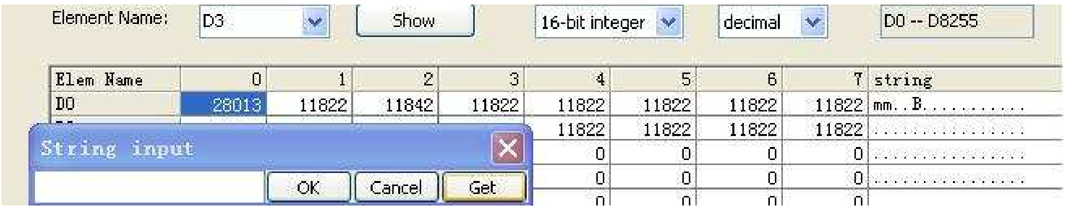


Hint:

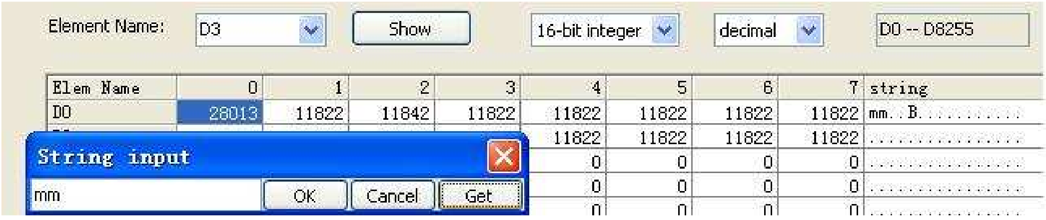
You can only enter the string into the last column of the soft component memory table. In other words, regardless of whether you enter the string directly or double-click the cell in the last column, the value entered will be displayed in the form of a string. At the same time, the input string will be automatically mapped to the values of the corresponding D component in the same row.

Reading the strings of the D component:

Again using the D0 component as an example, a string input dialog box will pop up when you double-click the cell with the D0 component. Pressing the "Read" button in the dialog box will show the corresponding string content of D0 in the editing area (as shown in the following two figures):



Click the "Read" button, as shown below:



Hint:

If either the high 8-bit value or the low 8-bit value of the D component is less than 32, the system will prompt "Existing characters cannot convert". This means that the character read is expressed in points.

Showing the numerical value data types:

The system can show many data types of the D component. One D component can show a single 16-bit integer, while two D components combined can show a 32-bit integer, a 32-bit floating point, or a 32-bit fixed point.

The display of the D component value:

In addition to a variety of data types, the display of the D component can also use the Decimal or Hex.

Other basic functions of the component memory table:

In addition to the specific functions above, the soft component memory table also has the following functions: copy, paste, cut, and delete. These functions will not be introduced here since they are similar to the edit function of symbol table .

Hint:

The paste function of the soft component memory table can only be used for the same type of data copies. In other words, the content of the string list cannot be copied to the value of the D component. The copying function cannot be realized between different data types.

PLC online control and monitor

This section mainly introduces some operations relating to communication between Autoshop background software and PLC, and how to monitor PLC programs by using background software.

Overview

After you've written a program and downloaded it to the device, you may perform program monitoring to ensure the accuracy of programming logics .Prior to executing program monitoring, please ensure a functioning communication between the program device and the PLC device. Also, the program in PLC must be in agreement with the current operating program.

Entering monitoring mode:

- Selecting [Debug/Monitor] on the menu or click on the monitor button （)in the toolbar. System will first detect the communication between Autoshop and PLC to ensure effective communication, and then verify the monitoring password to see if it has been set up in the PLC. Correct monitoring password must be entered when necessary to enter the monitoring mode. Otherwise, the program cannot be edited.

Exiting monitoring mode:

- When program monitoring needs to be terminated, select [Debug/Monitor] in the menu or click on the monitoring button () in the toolbar. At this time, the program will be switched from monitoring mode to editing mode.

Note:

1）System does not require the PLC hardware to be in a running state when using the monitoring function,. Nevertheless, if you would like to ensure the programming logic of the monitoring device to remain accurate, it is recommended to set up your device to maintain in a running state, or the debug results may not be accurate.

2）System will read and retrieve soft component values from the PLC device on a regular basis, and display the information in the program display window while monitoring. That means the monitoring results are not in real-time.Meanwhile, the PLC device's CPU loop-scan has a very short cycle, which is far shorter than the system reading cycle. Therefore, there will be delays between the monitored component value and the actual value. Understanding this point will help you in exercising programming logics and diagnosing the execution results.

3）n order to ensure the monitoring efficiency, system will only check the programming consistency in upper and lower devices when entering the monitor mode. Therefore, if the monitored PLC was switched during the process, the system will not recognize this operation. This will result in inconsistency between the monitored results and the actual operating results.Therefore, it is recommended not to perform such operation.

Start up PLC

Users can start up the PLC through the Autoshop software. After the PLC has been started up, it enters the operating state and start to execute user's logic control program.

Click on the "PLC/Start up" menu or pressing F5 key to execute this function, and it will appear in information display window:

PLC Termination

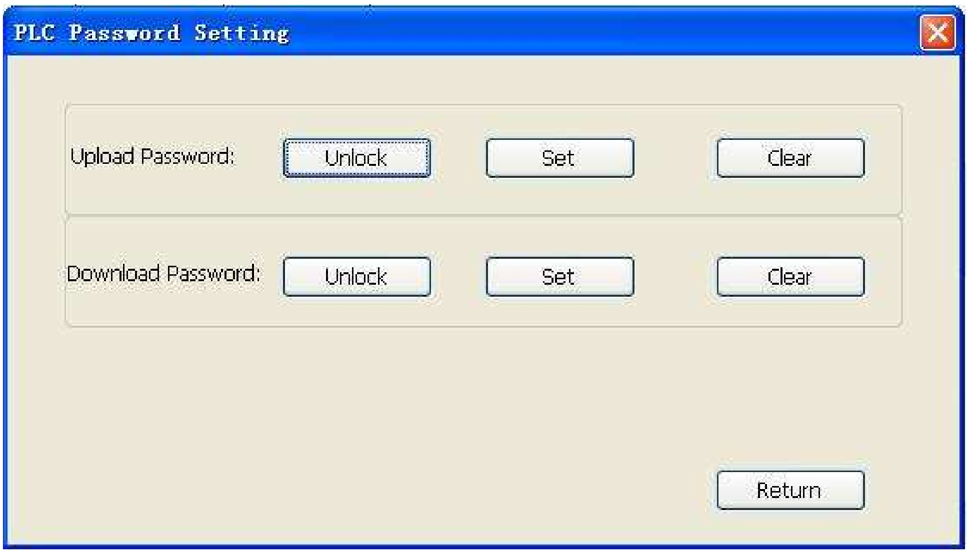
User can terminate the PLC with the Autoshop software. After PLC has been terminated, it also terminates the execution of user's logic control program.

"PLC/Stop" menu or pressing F6 key to execute this function, and it will appear in the information display window.

PLC encryption setting

The PLC encryption setting function is specifically designed for the purpose of PLC program uploads and downloads.

Clicking on the [PLC/Encryption setting] menu item, and the PLC encryption setting dialog box will pop up:



As illustrated above, the encryption setting contains two sections: upload password and download password. Due to the procedure of setting passwords for both sections are identical, only the operating procedure for setting upload password will be explained here:

Unlocking password

Click on the button of "Unlocking password" in the upload password setup screen. If user did not set up password for uploads in PLC setting previously, the message "No password has been setup." will pop up; on the other hand, the following dialog box will pop up:

After entering the correct password and clicking "OK", the message "Upload password unlocking command has been successfully executed" will appear in the information display window.When user is uploading programs, it is not necessary to perform upload password verification.

Setting up password

Click the "Setting up password" button in the upload password setup screen. If user did not set up upload password in PLC settings previously, a dialog box illustrated below will pop up:

As illustrated above, click the "OK" button after the new password has been entered and confirmed. At this time a message "upload password modification command has been successfully executed" will appear in the information display window.

If user already set up an upload password in PLC previously, the following dialog box will pop up:

Users need to enter the correct old password, and then enter a new password and click the "OK" button after confirm the new password. At this time, the message "upload password modification command has been successfully executed" will appear in the information display window.

Clearing password

Click on the "Clearing password" button in the upload password setup screen, and the "Password verification" dialog box will pop up:

If user did not set up upload password previously, a prompt dialog box box will pop up after clicking the "OK" button; on the other hand, after the correct upload password is entered and click on the "OK" button, the message "Upload password clearing command has been successfully executed" will appear in the information display window.

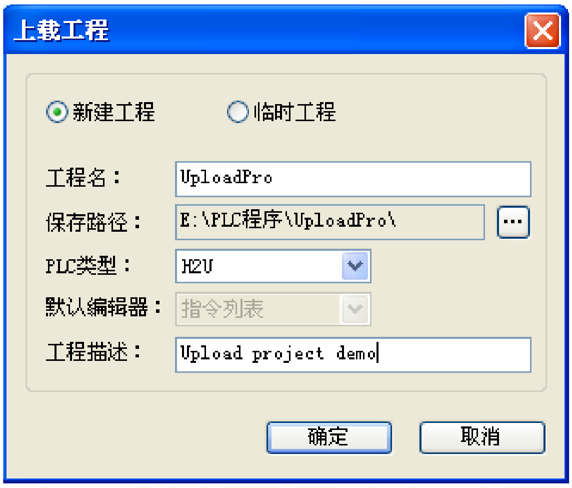
Download

The operation for downloading PLC programs has been explained in the Quick Start section.

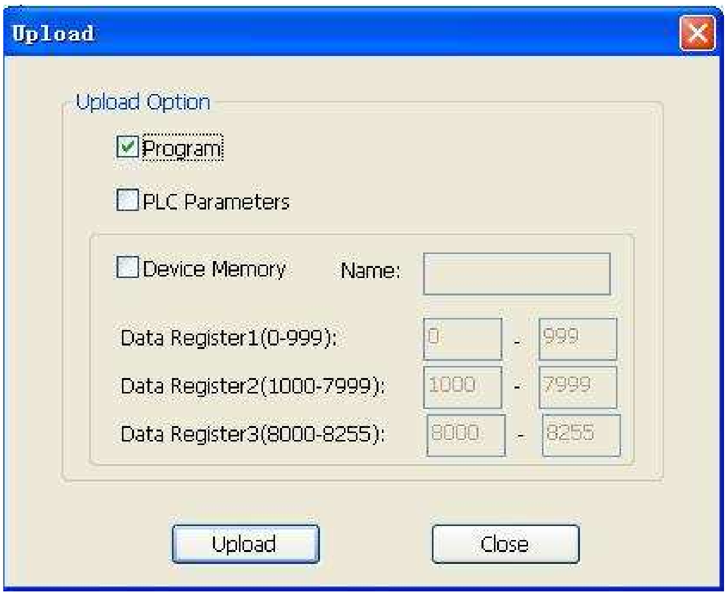
Upload

The upload function is used to upload and save applications, system parameters, and soft component memory to local computers as project files, and then to generate new projects.

To upload a project, you can click on [PLC/upload] in the menu or the upload button () in the toolbar. The following dialog box will then pop up:



As illustrated above, enter a project name; select a save path for the new project; click on the "OK" button after entered the project description. You will proceed to the next step for uploading projects as illustrated below:



As illustrated above: there are three upload options:

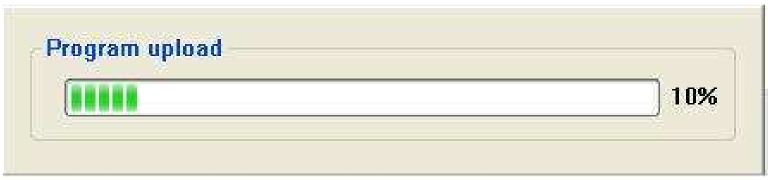
1£©Applications, which are user programs in the project. It includes the main, sub-, and interrupt programs.

2£©System parameters. Please refer to the parameter overview the parameter overview section in Quick Start.

3£©Soft component memory, which is the data uploads in the data storage component (D component).

As a demo, only "Application upload" is selected here. After clicking the "upload" button, should the PLC's upload password has been set up, the "Upload password verification" dialog box will pop up as illustrated:

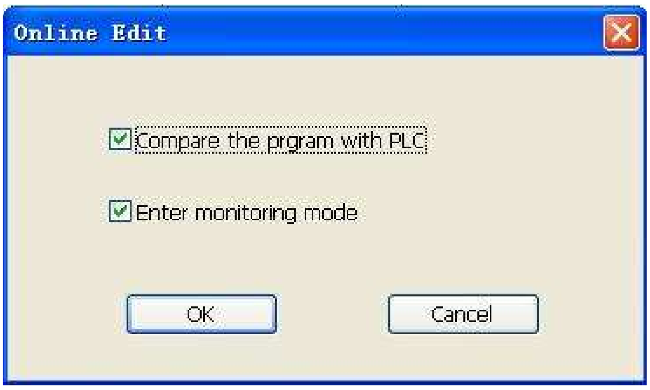
After entering the correct upload password and click on "OK", the last step of uploading programs will appear as illustrated below:



The uploading progress bar is illustrated above. After the upload is completed, Autoshop will automatically open the uploaded main program.

Online editing

Autoshop not only can monitor program, but also edit online.click the  button,the dialog Online edit will pop up:

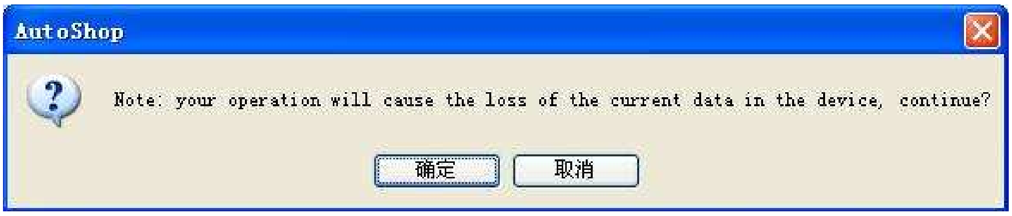


Modified current project and click OK, software compile automatic.After compile successful, if click download software will compared project current and this in PLC,download difference part.User needn't stop connect, the modified project will be run by PLC derectly.

Clear PLC program storage space

Clearing PLC program storage space means that all user programs in PLC will be cleared. Before clearing the programs, the PLC should first be terminated.

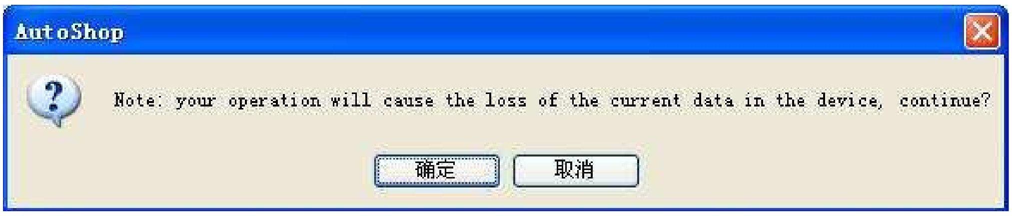
When clearing PLC programs, any other user programs will not be functioning except functions such as input and output, communication, and internal processing. This is very important, and please acts with cautions. When operating, the software will prompt a confirmation dialog box. Click the "OK" button to continue, or "Cancel" to exit.



Clearing digital data component storage space

Clearing digital data component storage space means that all data block settings in PLC will be cleared. PLC should be terminated prior to the execution of command.

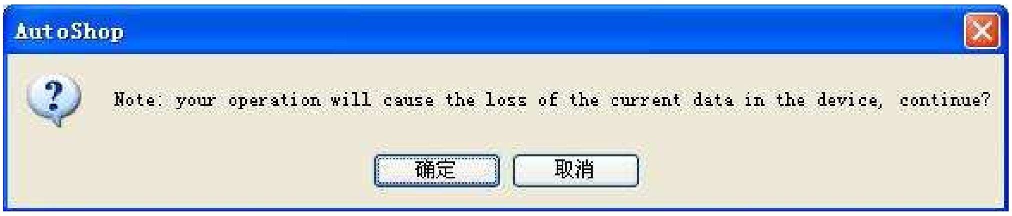
Clearing the data blocks in PLC will result in the pre-set value initialization and the D component will not be applied after PLC returns to operation. Please act with cautions. Prior to the execution, the software will prompt a confirmation dialog box. Click "OK" to continue, or "Cancel" to exit.



Clearing bit component storage space

Clearing PLC component storage space means that all component values in PLC will be cleared. The PLC should be terminated prior to the execution of this command.

Clearing the component values in PLC will result in abnormal PLC operation or operating data lost. Please act with cautions. Prior to the execution of the command, the software will prompt a pop up confirmation dialog box. Click "OK" to continue, or "Cancel" to exit.



Note:

The digital value, which is set as power failure benchmark, will also be cleared when command is executed.

Making dial-up connection with Modem

The purpose of Modem dial-up connection:

Use INOVANCE's Autoshop programming software to perform remote control over far end PLCs through any PC. It includes start-up/stop control, program upload and download, and etc.

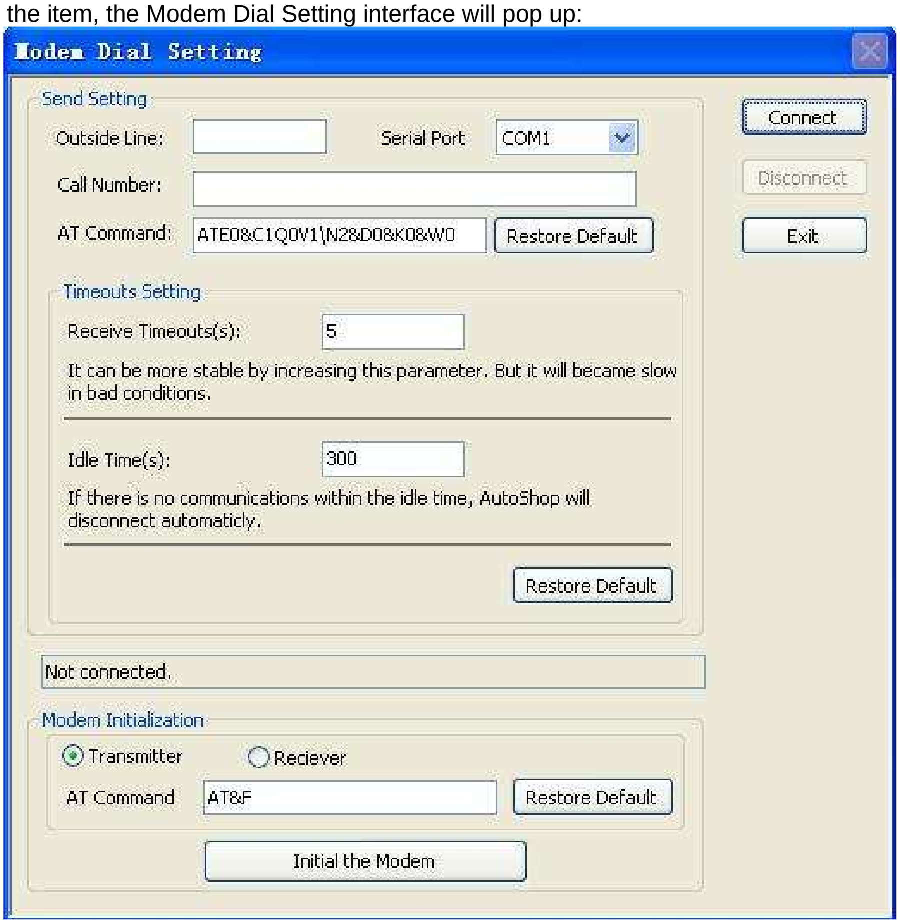
Principle of modem dial-up connection:

Two Modems are required: one at the PC end, and another at the PLC end.The modem at the PC end is connected to the PC's COM port and telephone line. The modem at the PLC end is connected to the PLC's COM port and telephone line as well. After the connection is successfully established, use the Autoshop software to perform dial-up procedure. If two modems are successfully connected, the PLC in the far end can be remotely controlled through the programming software in the PC.

Steps for dialling up with modem:

(1)Select [Modem Dial Setting] in the menu, as illustrated below:

(2)After clicking on the item, the Modem Dial Setting interface will pop up:

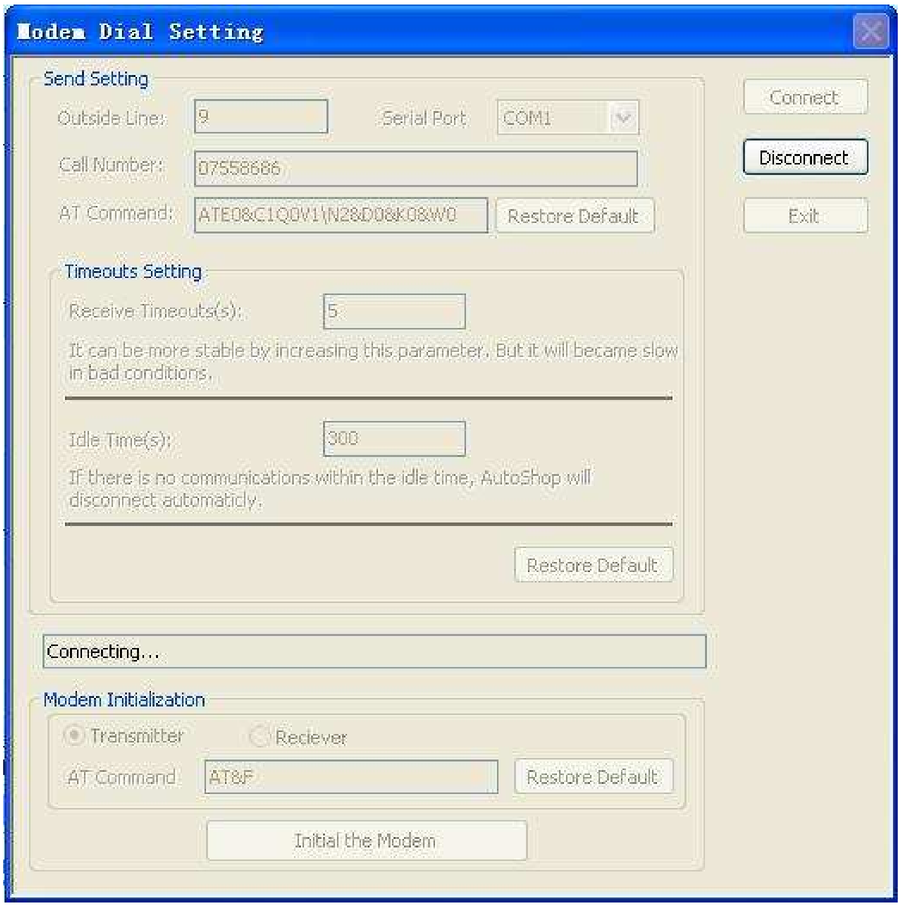


(3)Modem Initialization:

Connect the modem to the PC and click on the "Transmitter" or "Receiver" button to select the corresponding option.AT instruction's designation can be adjusted according to the actual modem model. Click the "Initial the modem" button at the end.

(4)Send Setting:

Select the port number used at the PC end from the Send Setting dialog box and enter the telephone number. If the modem is connecting to an outside line, the outside line number must be entered as well. AT instruction's designation can be adjusted according to the actual modem model. Click on the "connect" button, as illustrated below:



(5)If disconnect is desired after connection has been successfully established and have exited the dialog box, please re-open the dial-up dialog box and click on the "disconnect" button in step one.

(6)Timeouts Setting:

After verifying the wiring layout and yet the communication is still not established, you may consider to increase the "Receive Timeouts(s) " value in the [Timeouts setting] and then try to dial up again. This may compensate for the impact caused by unstable line condition.

SFC program monitoring

SFC program monitoring explains the procedures in writing soft component values and operating monitoring list in the SFC monitoring mode.

LD program monitoring

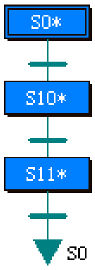
LD program monitoring explains the procedure in writing soft component values and operating the monitoring list in the LD monitor mode.

IL program monitoring

IL program monitoring explains the procedures in writing soft component values and operating the monitoring list in the IL monitoring mode.

Components can be monitored

Only the initial and general stepping sign components can be debugged.When the status of the corresponding S soft component is "ON", these components are displayed as accessible.



Writing in component values

When monitoring the program, in order to have the program running according to the specific logics programmed by users, values must be assigned to the components. This can be achieved by using the feature of writing in component values. The feature under the sequential function chart (SFC) has the same operating procedures as the ladder diagram. Please refer to Writing in LD section.

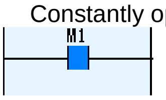
Adding components to the monitoring list

Under the SFC monitoring mode, the operating procedure for adding components to the monitoring list is the same as that in the LD mode. Please refer to the monitoring mode section in the LD chapter .

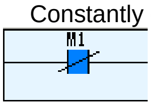
Components can be monitored

Following components can be monitored in the ladder diagram (which means that under the monitoring mode different values will be displayed accordingly):

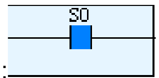
Constantly opened contact: when the soft component value is 1, it displays as "connected". See illustration:



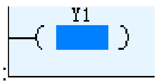
Constantly closed contact: when the soft component value is 0, it displays as "connected". See illustration:



Step contact: when soft component value is 1, it displays as "connected". See illustration:



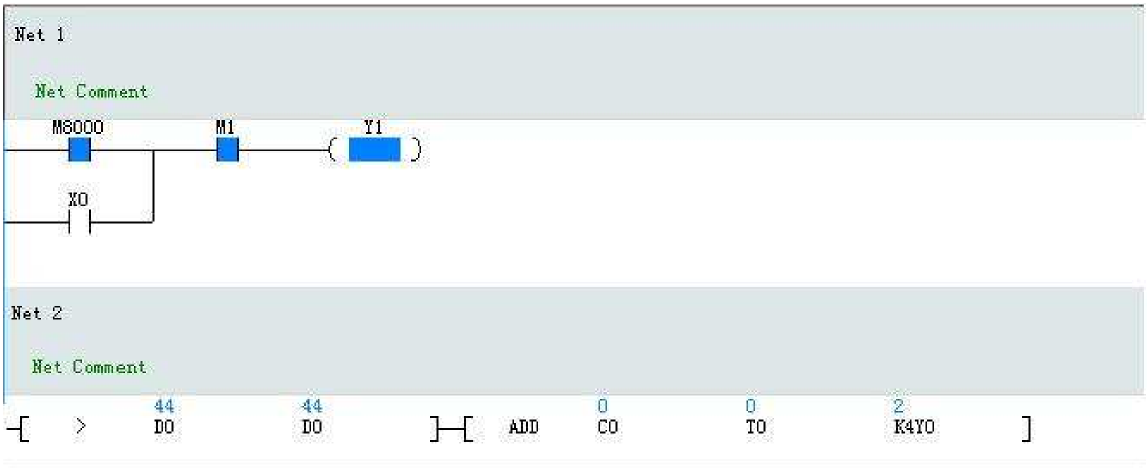
Coil: when soft component value is 1, it displays as "connected". See illustration:



Comparative contacts and application commands: normally this type of commands has multiple operands. Not only it covers the bit components, but also the data components. Therefore, the data components will also displays these values.

See illustration:

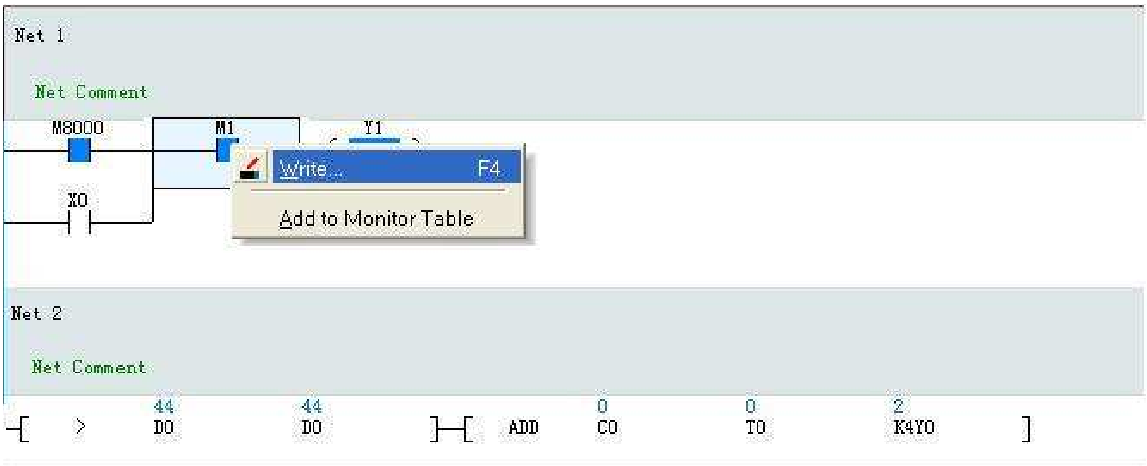
Please see illustration below for an application program that is currently being monitored:



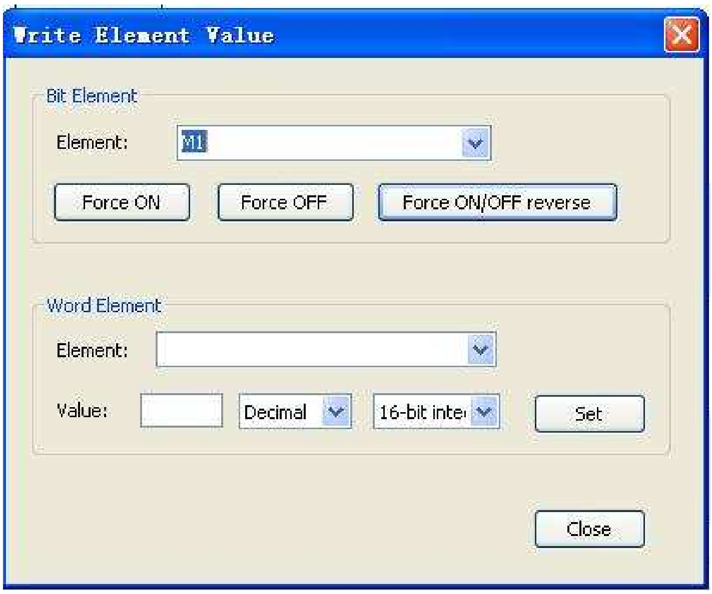
Writing in component values

In order to allow the program to function properly according to the logics designed by the users while monitoring the program, component values must be assigned to the components. This can be achieved by using the component value writing function: in order to write in the component values, the operating procedure is as follows:

Select a component that can be monitored, and select [Debug/Write] in the menu or click the "write" button( ). Or use right click and select "write" in the pop up menu as illustrated:



A dialog box for inputting component values will pop up after clicking the "write" button. Please see the illustration:

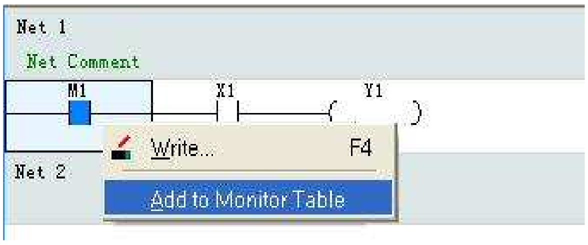


As illustrated above, there are two types of component inputs, bit-soft and byte-soft component inputs. As for bit soft component input, the forced operation over the component value can be set up by clicking on the "Imperative ON", "Imperative OFF", and "Imperative On\Off inverse" buttons. As for byte soft component input, first enter the desired values into the editing box, and then click on the "Set" button to perform editing over the byte-soft components.

Adding components to the monitoring list

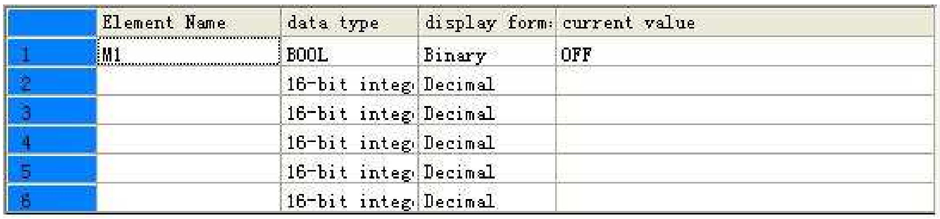
A monitoring list is specifically designed in the Autoshop to store the component status messages and provide descriptions for the monitoring list while in the monitoring mode. This has already been introduced in the Quick Start section and will not be repeated here. The operating procedures for adding components to the monitoring list will be demonstrated in this section.

Under the monitoring mode, select a component and right click to select "Add to the monitoring list" in popup menu. Please see illustration below:



A dialog box for adding component to the monitoring list will pop up as illustrated below:

Select the desired monitoring list name to be added, and click the "OK" button. Open the monitoring list, and you can see the components have already been added in. See illustration:

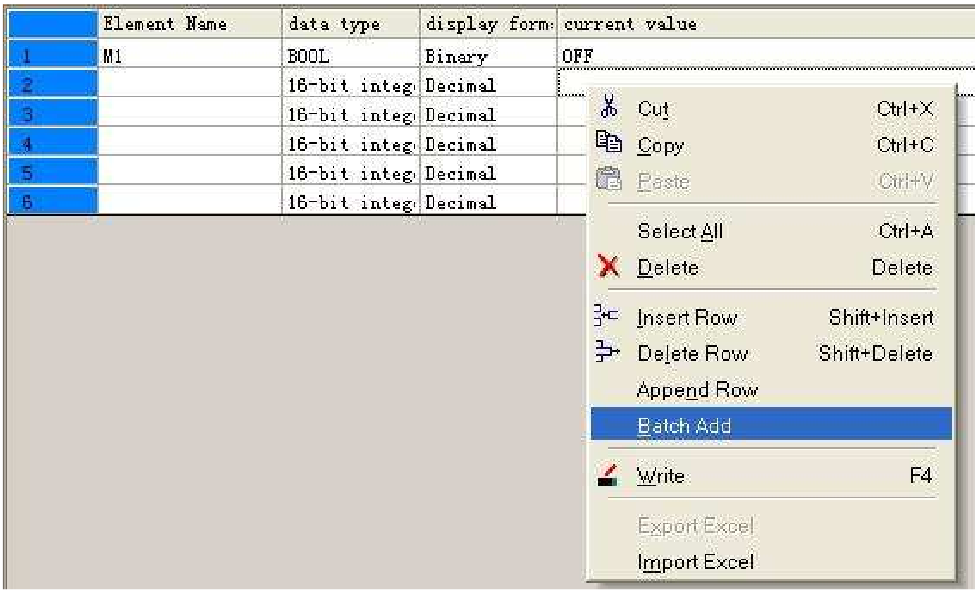


	Element Name	data type	display form	current value
1	M1	BOOL	Binary	OFF
2		16-bit integ	Decimal	
3		16-bit integ	Decimal	
4		16-bit integ	Decimal	
5		16-bit integ	Decimal	
6		16-bit integ	Decimal	

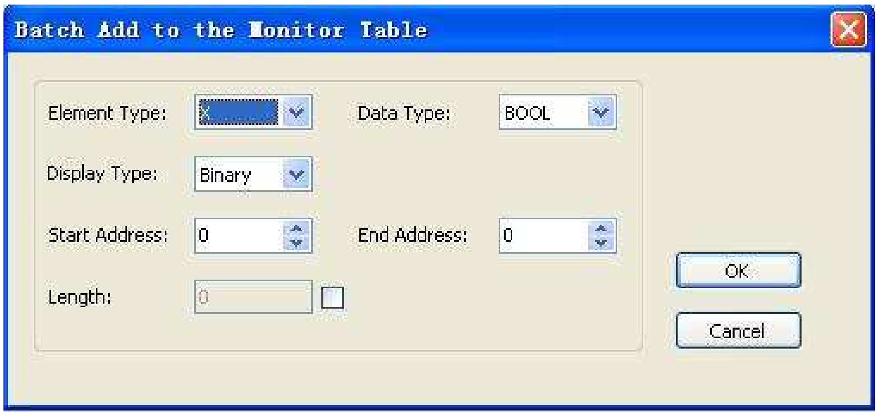
Adding batches of components to the monitoring list

In order to provide users the function of performing sequential soft components batch monitoring over multiple addresses under the monitoring mode, Autoshop has added the new feature of soft components batch monitoring.Operation procedures are as follows:

1.Open the monitoring list (under the monitoring mode), right click, and select the "add batch" menu item as illustrated:

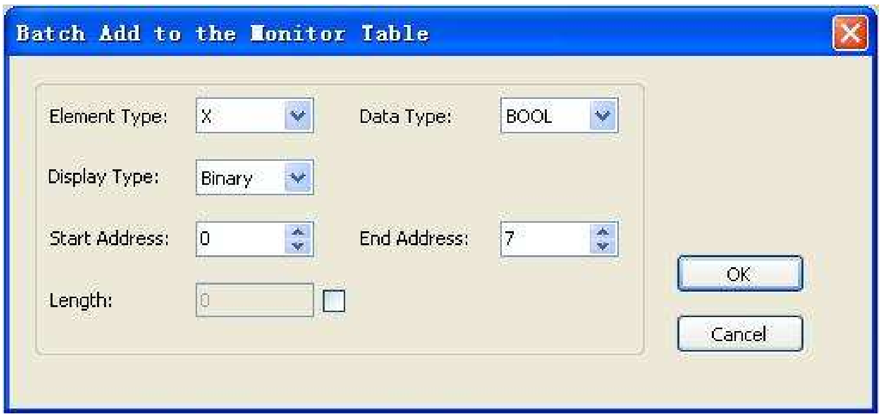


2.After completed the first step, the interface of adding soft component batch(es) will pop up, which is illustrated below:

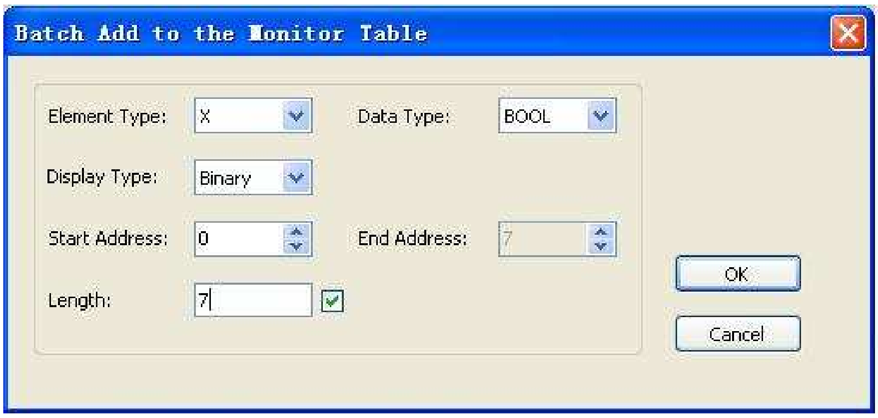


3.There are two ways of adding soft component batches:

a.Select the component's start and end addresses, and click the "OK" button to add the sequential soft components in multiple addresses into the monitoring list.



b.Enabling the length editing box (click the button next to the edit box), and entere the number of sequential soft component addresses to the length editing box. Click "OK".



See the following demonstration:

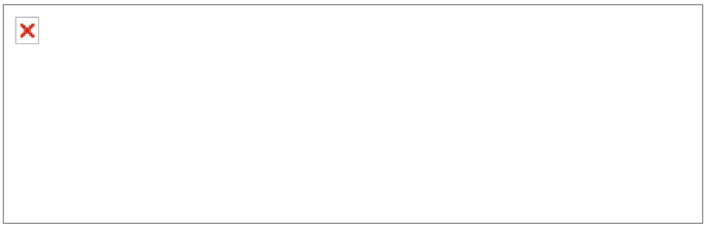
	Element Name	data type	display form	current value
1	M1	BOOL	Binary	OFF
2	X0	BOOL	Binary	OFF
3	X1	BOOL	Binary	OFF
4	X2	BOOL	Binary	OFF
5	X3	BOOL	Binary	OFF
6	X4	BOOL	Binary	OFF
7	X5	BOOL	Binary	OFF
8	X6	BOOL	Binary	OFF
9		16-bit integ	Decimal	
10		16-bit integ	Decimal	
11		16-bit integ	Decimal	
12		16-bit integ	Decimal	
13		16-bit integ	Decimal	

IL program monitoring

When comparing the SFC program and the ladder diagram program under the IL monitoring mode, the component's controlled state can only be represented in the monitoring list, which has a relatively simpler operating procedure. See illustration below:

After selecting a component, right click and select "added to monitor list" command in the pop menu. See illustration:

A dialog box for adding component to monitoring list will pop up. Click "OK" after selected the monitoring list name as illustrated below:



At this stage the component information has been added to the monitoring list, and the information can be found when opening the monitoring list. The monitoring feature of the instruction list program is in fact the equivalence to the "Adding component to monitor list" feature in the ladder diagram and SFC programs. The operating procedures are essentially identical.

Soft component description

It includes soft component types, which are supported by INOVANCE PLC, and their function descriptions.

Types of software components

INOVANCE PLC is supported by the soft component types in the following table:

SN	Component Type	Features and classification
1	Input Relay X	PLC hardware corresponding bit component digital inputs
2	Output Relay Y	PLC control output corresponding to digital components
3	Intermediate relay M	Common intermediate relay M-bit components; system special relay M-bit components
4	State relay S	Step control components with status flag
5	Timer T	With 1ms, 10ms, 100ms step of 16bit timers
6	Counter C	With 16bit/32bit up / down type counter, high-speed counter, single / duplex various counter
7	Data register D	Data register D; data register indirect addressing V, Z, D register file
8	Pointer P¡¢I	Jump pointer P, subroutine pointer P, interrupt subroutine I, a high-speed input, timing, counting and other interruptions
9	Constant K¡¢H	Binary, decimal, hexadecimal, floating point, etc.

Input relay X

The input relay X represents the PLC status of the external input signal components. And it can get through the input port to detect the external signal status. 0 is for external signal open circuit, and 1 closed.

It cannot modify the state input relays in the way of program instructions. Contact signal (normally open, normally closed type) program can be used an unlimited amount of times by the user.

The number of Relay signals is X0, X1¡X7, X10, and X11 and so on. The serial number is in octal numbers. Controller counter signal, external interrupt, pulse catch functions through the input port X0¡«X7.

Model	Input	Output
H2U-1616MR/T	X000-X017	Y000-Y017
H2U-2416MR/T	X000-X027	Y000-Y017
H2U-3624MR/T	X000-X044	Y000-Y027
H2U-3624MTQ	X000-X044	Y000-Y027
H2U-3232MR/T	X000-X037	Y000-Y037
H2U-4040MR/T	X000-X047	Y000-Y047
H2U-6464MR	X000-X077	Y000-Y077

Output relay Y

Output relay is directly related to the external user to control the hardware port of the software component. It corresponds to the physical output port of PLC. The component status of relay Y will be sent to the state of the hardware port on the PLC. 0 indicates that the output port is open, and 1 closed.

The number of relay Y is Y0, Y1,...Y7, Y10, Y11 and so on. The number sequence is in octal numbers. Relay device in the user program can be used an unlimited number of times.

The hardware can be divided into the following categories: relay type, transistor type, solid state relay type, etc. according to different output devices. If it has the output expansion module port, relay Y will be numbered sequentially from the main module.

Auxiliary Relay M

Auxiliary Relay M components is used as an intermediate variable during the execution of a program, as auxiliary relays in the practical power control system which is used to transfer the status messages. It can use the word variable formed by M variables. M variables is not directly linked with any external ports, but it can contact with the outside world by the manners of copying X to M or M to Y through the program coding. A variable M can be used repeatedly.

Auxiliary Relay M can be identified with the symbols of M0, M1,...,M8255. The numbering system is numbered by 10 hex. The variables that are more than M8000 are the system-specific variables,which is used to interact with the PLC user program with the system status; part of the M variables have the feature of power-saving,

Total number of M	General	Latched	Latched dedicated	Special
3082 points	M0-M499 (384 points)Tip1	M500-M1023 (524 points) Tip2	M1024-M3071 (2048 points) Tip3	M8000-M8255 (256 points)

Tip1: Non-latched area. The non-latched area can be changed to a latched area with the parameter setting.

Tip2: Latched area. The latched area can be changed to a non-latched area with the parameter setting.

Tip3: Cannot change the characteristics in order to maintain the power-off by parameter setting.

The regional distribution of the generally used auxiliary relays and the auxiliary relays that are latched in the programmable controller can be adjusted by adjusting the settings in the parameter.

Programmable controller has a large number of special auxiliary relays. Each one of them has their specific functions which can be categorized into the following types:

1) The special auxiliary relays used for contacts. For instance:

M8000: Operating monitor (connected in operation). It is commonly used before the command signal execution.

M8002: The initial pulse (only connect shortly at the beginning of operation), it is commonly used as the initialization command.

M8012: 100ms clock pulse. It is used to generate a signal at during regular interval flips.

2) Coil-driven special auxiliary relays provide driven coils for user programs, and it is used to control the operating status and the status of execution of the PLC. For instance:

M8030 : The command for battery lighting and polar tube lighting.

M8033 : Continue exporting when stopping

M8034 : Total ban on export

M8039 : Constant Scanning

Note:

M component is effective when there is a driver and two cases after the execution of the END command; users cannot use the special auxiliary relays that have not yet been defined.

State relays S

State relays S is used to design and handle step procedures, control the transfer steps of the state S by STL step instructions, and simplify programming. If there is no way of using STL programming, S can be used as M. S variables are identified with S1...S999 and so on. The serial number is a decimal number. Part of the S variable has the function of power-down save.
See the following table:

General use			**Latched**			**Alarm Used**
S0-S499 (500) Tip1	S0-S9 (10)	S0-S9 (10)	S200-S899 (400) Tip2	-	-	S900-S999 (100)Tip3

Tip1: No latched area. Parameter settings can be changed through the power outage to maintain the

leading city.

Tip2: Latched area. Parameters can be changed by setting the leading city of non-latched.

Tip3: Latched features. It can not be changed by setting the parameters.

Timer T

The timer is used to perform the timing function. Each timer contains coils, contacts, and counting time value register. When the coil sounds (with sufficient power), the timer starts timing. If the timer's registered value reaches the preset value, the contacts activate, and other contacts (NO contacts) are closed, while b contacts (NC contacts) disconnect. If the coil power shuts off (insufficient power), the contacts will restore to their initial states and the value will automatically be cleared. Some timers have the feature of accumulation and shut-down.After a restart, it will even keep the value before the shut--down.

Timer T is expressed by the symbol of T0, T1, ... T255. **I**ts serial number is in 10 decimals.

Timers have different timing steps. For instance, 1ms, 10ms, 100ms, and etc. See the following:

100ms Type 0.1~3276.7s	100ms type 0.01~327.67s	10ms type 0.01~327.67s	1ms type 0.001~32.767s	100ms cumulative 0.1 ~ 3276.7s
T0 ~ T199 200 points are generally used in the procedures of T192 ~ T199	-	T200 ~ T245 46 points	T246 ~T249 4 locations to keep with interruption	T250 ~ T255 6 points remained

Tip:

The timer number is not for the timer, and it can be used as a data register to save values.

Counter C

The counter is used to complete a counting function, and each counter contains a coil, contacts, and time value register. When the counter coil drives a signal from the OFF to ON, the value of the counter is increased by 1. If the time value reaches the preset time value, the contacts act, Contact a (Contact NO) closed, Contact b (Contact NC) disconnected. If the timer value is cleared, output of Contact a is broken, Contact b (Contact NC) closed. With power-down to maintain a cumulative and other characteristics, after re-powering, some of the counters can maintain the value before powering-down.

Counters are identified by C0, C1, ..., C255, ordered by 10 hexadecimal numbers.

Counters have the width of 16bit and 32bit. There are the single-way counting type, change counting type, bipolar counting type, etc. Some of the counters have the option to maintain values when powered-down. Select the appropriate counters according to need when using.

16-bit cis/ counter counting from 0 to 32,767		**32-bit cis / counter counting from -2,147,483,648~+2,147483647**		
General	Latched	Latched dedicated	Special	High-speed counter
C0~C99 (100point) #1	C100~C199 (100point) #2	C200~C219 (20point) #1	C220~C234 (15point) #3	C235~C255 (21point) #1, #2

1 Non-latched area. The non-latched area can be changed to a latched area with parameter setting.

2 Latched area. The latched area can be changed to a non-latched area with parameter setting.

3 You cannot choose the option to maintain a value when powered-down by parameter setting.

Tip:

A counter number not used as a counter, can be used as a data register to store data.

Register D

Data Register D

The register is used for data computation and storage ¨C items such as timers, counters, and analog parameters of the operation. Each register is 16 bits wide. If using the 32-bit instructions, it will be composed of adjacent registers to use as a 32-bit register.The address low byte is low byte, and the opposite is high byte.

In H2U series PLC's instructions, the majority of the data is carried out by a number of processed symbols.For the 16-bit register, bit-15 is the sign bit (0 for positive numbers; 1 for negative numbers).As for the 32-bit register, high byte bit-15 is the sign bit.The range of values is : from (-32 , 768) to (+32, 767).

When the data need to be addressed as 32-bit, it can be as two adjacent D registers composed of 32- bit double word. For example, when we need to visit D100 in a 32-bit format, the high-address register D101 is high byte, and the high byte of 15 is a two-word sign bit.The following values can be handled: -2 , 147 , 483 , 648 to 2 ,147 , 483 , 647.

The register is identified with D0, D1, ..., D9, 999, carried out according to the decimal number.

General	**Latched**	**Latched dedicated**		**Specialty**	**Designation**
		General	**(File use)**		
D0~D199 (200) Tip1	D200~D511 (312) Tip2	D512~D7999 (7488) Tip3	After data register D1000, it is the register to remain for file	D8000~D8255 (256)	V0~V31 Z0~Z31 (64) Tip3

			Register		

Tip1: No latched area.Parameters can be changed by setting the field latched.

Tip2: Latched area.Parameters can be changed by setting the field of non-latched.

Tip3: It cannot be changed by setting the parameters' latched features.

Index Register V, Z

Index register V, Z is the same as the common data register, which is for the numerical data to read and write a 16-bit data register. It is a total of 64: V0~V31, Z0~Z31.

The index register has the same use as the common data register, and it also can be used with other numbers or values of soft components.But we need to be aware that LD, AND, OUT, and other basic sequential control commands or a step ladder program cannot be a soft component number used in combination with the index register.

File register D

After data register D1000, it is the register to remain. Through parameter settings, it can be specified from 1 to 14 blocks for the backup file. But a record for each additional block of 500 steps of the procedure would reduce the storage area. When part of the D1000 is set to file register, the rest can still maintain a register used as a general one.

Pointer L, I

Point L is applied as the entry address of jump routines and as well as the label of subroutine starting addresses; Pointer I is applied as the label of starting address of interrupt routines, and its codes are allocated to decimal digit as the table below shows:

Dots used by branches	**Special instructions**	**Dots used by input interrupt**	**Dots used by timer interrupt**	**Dots used by high-speed counter interrupt**
L0~L127 127 dots together	During editing programs, operands of the CALL instruction are subroutine names. While subroutine names achieve addressing through corresponding L designators when processed inside PLC. Thus, it should be noted that the sum of the subroutine number and the number of	I00x(X0) II10x(X1) I20x(X2) I30x(X3) I40x(X4) I50x(X5) x=0 rising edge interrupt x=1 trailing edge	I600 I700 I800 dots together	I010 I040 I020 I050 I030 I060 6 dots together

	routine jump designators cannot be greater than 127.	interrupt 12 dots together		

Constant K,H,E

According to different application and purposes, H2U series programmable controller uses 5 types of values. Their role and functions as follows:

Type	Application Notes in Programming
Decimal (DEC)	The set value of timer and counter (K is a constant) The number of Auxiliary Relay(M), Timer(T), Counter(C), Status(S) and so on (the number of soft component) The value and command action in the Operand, which are applied (K is a constant)
Hexadecimal number (HEX)	As with the 10 decimal number, it is applied in the operand and the specific actions in the application commands.
Binary (BIN)	Using decimal number or hexadecimal number to design the value of the timer, counter or data register. However, in the internal programmable controller, these figures are dealt with binary numbers. Moreover, when monitoring external devices，these soft components will be converted to a decimal number automatically (16 hex can be converted as well)
Octal (OTC)	Using 8 hex values to distribute the soft component number of Input relay and output relay. Use the binary values of [0-7, 10-17, ... 70-77, 100-107]. [8, 9] does not exist in the 8 hexadecimal number.
BCD	BCD is a way of using 4-bit binary to represent decimal values. The processing of these numbers is simple. Thus, it can be used in the digital switch of BCD output format and the display control of seven segments.
BIN float	Programmable controller has the function of high-precision floating point capabilities. In the center, use binary (BIN) floating-point to conduct floating-point operations
Decimal floating point	Decimal floating-point value is only used for monitoring and improving readability.

Constant K

[K] is the symbol that expresses the 10 decimal integer. It is used to set the value of the timer, the counter, and the value in the operand. In the 16bit commands, the constant K ranges from -32768 to 32767; in the 32bit commands, the constant K ranges from -2,47,483,648 ~ 2,147,483,647.

Constant H

[H] is the symbol that expresses the 16 decimal integer. It is used to set the values in the application command operand. Constant H ranges from 0000 ~ FFFF;in the 32bit commands, the constant K ranges from 0000,0000 to FFFF, FFFF.

Constant E

[E] is the representation of a 32-bit floating point. It is used to set the values in the application command operand.

Instruction list

In instruction list, the instructions are divided into basic instruction, step ladder diagram instruction, program flow instruction, transmission and comparison instruction, arithmetic instruction, cyclic shift instruction, data processing instruction, high-speed processing instruction, easy instruction, external device IO instruction, external device SER instruction, floating calculation instruction, locating instruction, timer calculation instruction, external device instruction, and contact comparison instruction, which is described respectively.

Basic Instruction

Instruction Type	Instruction	Function Description
Basic Instruction	LD	Initial logical operation contact type NO (normally open)
	LDI	Initial logical operation contact type NC (normally closed)
	OUT	Coil drive
	AND	Serial connection of NO (normally open) contacts
	ANI	Serial connection of NC (normally closed) contacts
	OR	Parallel connection of NO (normally open) contacts
	ORI	Parallel connection of NC (normally closed) contacts
	LDP	Initial logical operation - Rising edge pulse
	LDF	Initial logical operation - Falling edge pulse
	ANDP	Serial connection of Rising edge pulse
	ANDF	Serial connection of Falling edge pulse
	ORP	Parallel connection of Rising edge pulse
	ORF	Parallel connection of Falling edge pulse
	ORB	Parallel connection of multiple contact circuits
	ANB	Serial connection of multiple parallel circuits
	MPS	Stores the current result of the internal PLC operations
	MRD	Reads the current result of the internal PLC operations
	MPP	Recalls and removes the currently stored result

	MC	Start of a master control block
	MCR	End of a master control block
	INV	Inverts the current result of the internal PLC operations
	PLS	Rising edge pulse
	PLF	Falling edge pulse
	SET	Coil set
	RST	Coil reset

Step ladder instruction

Instruction Type	Instruction	Function Description
Step ladder instruction	STL	Start of Step ladder instruction
	RET	End of Step ladder instruction

Program control instruction

Instruction Type	Instruction	Function Description
Program control instruction	CJ	Conditional jump
	CJP	Conditional jump(pulse type)
	CJEND	Jump to the end of program
	CJPEND	Jump to the end of program (pulse type)
	CALL	Subroutine call
	CALLP	Subroutine call(pulse type)
	EI	Interruption permissible
	DI	Interruption forbidden
	WDT	Monitor timer
	WDTP	Monitor timer(pulse type)
	FOR	Start of loop range
	NEXT	End of loop range

Move and Compare Instruction

Instruction Type	Instruction	Function Description
	CMP	Comparison of 16-bit data
	CMPP	Comparison of 16-bit data (pulse type)
	DCMP	Comparison of 32-bit data
	DCMPP	Comparison of 32-bit data (pulse type)

Move and Compare Instruction	ZCP	Comparison of 16-bit zone
	ZCPP	Comparison of 16-bit zone (pulse type)
	DZCP	Comparison of 32-bit zone
	DZCPP	Comparison of 32-bit zone (pulse type)
	MOV	Move of 16-bit data
	MOVP	Move of 16-bit data (pulse type)
	DMOV	Move of 32-bit data
	DMOVP	Move of 32-bit data (pulse type)
	SMOV	Shift move
	SMOVP	Shift move (pulse type)
	CML	Compliment move of 16-bit data
	CMLP	Compliment move of 16-bit data (pulse type)
	DCML	Compliment move of 32-bit data
	DCMLP	Compliment move of 32-bit data (pulse type)
	BMOV	Block move
	BMOVP	Block move (pulse type)
	FMOV	Fill move of 16-bit data
	FMOVP	Fill move of 16-bit data (pulse type)
	DFMOV	Fill move of 32-bit data
	DFMOVP	Fill move of 32-bit data (pulse type)
	XCH	Exchange of 16-bit data
	XCHP	Exchange of 16-bit data (pulse type)
	DXCH	Exchange of 32-bit data
	DXCHP	Exchange of 32-bit data (pulse type)
	BCD	BCD exchange of 16-bit data
	BCDP	BCD exchange of 16-bit data (pulse type)
	DBCD	BCD exchange of 32-bit data
	DBCDP	BCD exchange of 32-bit data (pulse type)
	BIN	BIN exchange of 16-bit data
	BINP	BIN exchange of 16-bit data (pulse type)
	DBIN	BIN exchange of 32-bit data
	DBINP	BIN exchange of 32-bit data (pulse type)

Arithmetic Operation Instruction

Instruction Type	Instruction	Function Description
Arithmetic Operation Instruction	ADD	Addition of 16 bit integers
	ADDP	Addition of 16 bit integers (pulse type)
	DADD	Addition of 32 bit integers
	DADDP	Addition of 32 bit integers (pulse type)
	SUB	Subtraction of 16 bit integers
	SUBP	Subtraction of 16 bit integers (pulse type)
	DSUB	Subtraction of 32 bit integers
	DSUBP	Subtraction of 32 bit integers
	MUL	Multiplication of 16 bit integers
	MULP	Multiplication of 16 bit integers (pulse type)
	DMUL	Multiplication of 32 bit integers
	DMULP	Multiplication of 32 bit integers (pulse type)
	DIV	Division of 16 bit integers
	DIVP	Division of 16 bit integers (pulse type)
	DDIV	Division of 32 bit integers
	DDIVP	Division of 32 bit integers (pulse type)
	INC	Add 1 to 16 bit integer
	INCP	Add 1 to 16 bit integer (pulse type)
	DINC	Add 1 to 32 bit integer
	DINCP	Add 1 to 32 bit integer (pulse type)
	DEC	Subtract 1 from 16 bit integer
	DECP	Subtract 1 from 16 bit integer (pulse type)
	DDEC	Subtract 1 from 32 bit integer
	DDECP	Subtract 1 from 32 bit integer (pulse type)
	WAND	Logical word and
	WANDP	Logical word and (pulse type)
	DWAND	Logical double word and
	DWANDP	Logical double word and (pulse type)
	WOR	Logical word or
	WORP	Logical word or (pulse type)
	DOR	Logical double word or
	DORP	Logical double word or (pulse type)
	WXOR	Logical word exclusive or
	WXORP	Logical word exclusive or (pulse type)
	DXOR	Logical double word exclusive or
	DXORP	Logical double word exclusive or (pulse type)
	NEG	16 bit negation instruction
	NEGP	16 bit negation instruction (pulse type)
	DNEG	32 bit negation instruction
	DNEGP	32 bit negation instruction (pulse type)

Rotation and Shift Instruction

Instruction Type	Instruction	Function Description
	ROR	Word Rotation right

Rotation and Shift Instruction	RORP	Word Rotation right（pulse type）
	DROR	Double Word Rotation right
	DRORP	Double Word Rotation right（pulse type）
	ROL	Word Rotation left
	ROLP	Word Rotation left（pulse type）
	DROL	Double Word Rotation left
	DROLP	Double Word Rotation left（pulse type）
	RCR	Word Rotation right with carry
	RCRP	Word Rotation right with carry（pulse type）
	DRCR	Double Word Rotation right with carry
	DRCRP	Double Word Rotation right with carry（pulse type）
	RCL	Word Rotation left with carry
	RCLP	Word Rotation left with carry（pulse type）
	DRCL	Double Word Rotation left with carry
	DRCLP	Double Word Rotation left with carry（pulse type）
	SFTR	Bit shift right
	SFTRP	Bit shift right（pulse type）
	SFTL	Bit shift left
	SFTLP	Bit shift left（pulse type）
	WSFR	Word shift right
	WSFRP	Word shift right（pulse type）
	WSFL	Word shift left
	WSFLP	Word shift left(pulse type)
	SFWR	Shift register write
	SFWRP	Shift register write（pulse type）
	SFRD	Shift register read
	SFRDP	Shift register read（pulse type）

Data Operation Instruction

Instruction Type	Instruction	Function Description
	ZRST	Zone reset
	ZRSTP	Zone reset (pulse type)
	DECO	Decode
	DECOP	Decode (pulse type)
	ENCO	Encode
	ENCOP	Encode（pulse type）
	SUM	Sum of active bits (16 bit)
	SUMP	Sum of active bits (16 bit, pulse type)
	DSUM	Sum of active bits (32 bit)
	DSUMP	Sum of active bits (32 bit, pulse type)
	BON	Check specified bit status (16 bit)
	BONP	Check specified bit status (16 bit, pulse type)
	DBON	Check specified bit status (32 bit)
	DBONP	Check specified bit status (32 bit, pulse type)
	MEAN	Mean (16 bit)
	MEANP	Mean (16 bit, pulse type)

Data Operation Instruction	DMEAN	Mean (32 bit)
	DMEANP	Mean (32 bit, pulse type)
	ANS	Timed annunciator set
	ANR	Annunciator reset
	ANRP	Annunciator reset (pulse type)
	SQR	Square root of 16 bit integer
	SQRP	Square root of 16 bit integer (pulse type)
	DSQR	Square root of 32 bit integer
	DSQRP	Square root of 32 bit integer (pulse type)
	WANDP	Logical word and (pulse type)
	FLT	Integer word to binary floating point conversion
	FLTP	Integer word to binary floating point conversion (pulse type)
	DFLT	Integer double word to binary floating point conversion
	DFLTP	Integer double word to binary floating point conversion (pulse type)
	SWAP	Word swap
	SWAPP	Word swap (pulse type)
	DSWAP	Double word swap
	DSWAPP	Double word swap (pulse type)

High-speed Processing Instruction

Instruction Type	Instruction	Function Description
High-speed Processing Instruction	REF	Input/output refresh
	REFP	Input/output refresh (pulse type)
	REFF	Filter adjust
	REFFP	Filter adjust (pulse type)
	MTR	Input matrix
	DHSCR	High speed counter reset
	DHSCS	High speed counter set
	DHSZ	High speed zone compare
	SPD	Speed detection
	PLSY	16 bit pulse output
	DPLSY	32 bit pulse output
	PWM	PWM(pulse width modulation)
	PLSR	16 bit ramp pulse output
	DPLSR	32 bit ramp pulse output

Handy Instruction

Instruction Type	Instruction	Function Description
	IST	Initial state
	SER	Search a 16-bit data
	SERP	Search a 16-bit data (pulse type)
	DSER	Search a 32-bit data

Handy Instruction	DSERP	Search a 32-bit data (pulse type)
	ABSD	Absolute drum sequencer (16 bit)
	DABSD	Absolute drum sequencer (32 bit)
	INCD	Incremental drum sequencer (16 bit)
	TTMR	Teaching timer
	STMR	Special timer
	ALT	Alternate state
	ALTP	Alternate state (pulse type)
	RAMP	Ramp variable value
	ROTC	Rotary table control
	SORT	Sort tabulated data

External IO Instruction

Instruction Type	Instruction	Function Description
External IO Instruction	TKY	Ten key input
	DTKY	Ten key input (32 bit operation)
	HKY	Hexadecimal key input
	DHKY	Hexadecimal key input（32 bit operation）
	DSW	Digital switch
	SEGD	Seven segment decoder
	SEGDP	Seven segment decoder (pulse type)
	SEGL	Seven segment with latch
	ARWS	Arrow switch
	ASC	ASCII code conversio
	PR	Print ASCII code
	FROM	Read 16-bit data from buffer memories of special function block
	FROMP	Read 16-bit data from buffer memories of special function block (pulse type)
	DFROM	Read 32-bit data from buffer memories of special function block
	DFROMP	Read 32-bit data from buffer memories of special function block (pulse type)
	TO	Write 16-bit data to buffer memories of special function block
	TOP	Write 16-bit data to buffer memories of special function block (pulse type)
	DTO	Write 32-bit data to buffer memories of special function block
	DTOP	Write 32-bit data to buffer memories of special function block (pulse type)

External SER Device Instruction

Instruction Type	Instruction	Function Description
	RS	Serial communication instruction
	MODBUS	MODBUS communication instruction
	CANTX	CAN transmission instruction
	CANRX	CAN receive instruction

External SER Device Instruction	PRUN	16 bit octal bit transmission
	PRUNP	16 bit octal bit transmission (pulse type)
	DPRUN	32 bit octal bit transmission
	DPRUNP	32 bit octal bit transmission (pulse type)
	ASCI	Convert HEX data to ASCII
	ASCIP	Convert HEX data to ASCII (pulse type)
	HEX	Convert ASCII data to HEX
	HEXP	Convert ASCII data to HEX (pulse type)
	CCD	Check parity code
	CCDP	Check parity code (pulse type)
	PID	PID control loop

Floating point operation instruction

Instruction Type	Instruction	Function Description
Floating point operation instruction	DECMP	Binary floating point data compare
	DECMPP	Binary floating point data compare(pulse type)
	DEZCP	Binary floating point zone compare
	DEZCPP	Binary floating point zone compare (pulse type)
	DEBCD	Binary to BCD floating point data conversion
	DEBCDP	Binary to BCD floating point data conversion (pulse type)
	DEBIN	BCD to Binary floating point data conversion
	DEBINP	BCD to Binary floating point data conversion (pulse type)
	DEADD	Binary floating point addition
	DEADDP	Binary floating point addition (pulse type)
	DESUB	Binary floating point subtraction
	DESUBP	Binary floating point subtraction (pulse type)
	DEMUL	Binary floating point multiplication
	DEMULP	Binary floating point multiplication (pulse type)
	DEDIV	Binary floating point division
	DEDIVP	Binary floating point division (pulse type)
	DESQR	Binary floating point square root
	DESQRP	Binary floating point square root (pulse type)
	INT	16-bit binary floating point to integer
	INTP	16-bit binary floating point to integer (pulse type)
	DINT	32-bit binary floating point to integer
	DINTP	32-bit binary floating point to integer (pulse type)
	DSIN	Floating point Sin operation
	DSINP	Floating point Sin operation (pulse type)
	DCOS	Floating point Cosine operation
	DCOSP	Floating point Cosine operation (pulse type)
	DTAN	Floating point Tangent operation
	DTANP	Floating point Tangent operation (pulse type)

Floating point operation instruction (only for special version)

Instruction Type	Instruction	Function Description
Floating point operation instruction	DEMOV	Binary floating point data move
	DEMOVP	Binary floating point data move (pulse type)
	DEXP	Binary floating point exponential operation
	DEXPP	Binary floating point exponential operation (pulse type)
	DLOGE	Binary floating point Natural Logarithm operation
	DLOGEP	Binary floating point Natural Logarithm Operation (pulse type)
	DLOG	Binary floating point Logarithm operation
	DLOGP	Binary floating point Logarithm operation (pulse type)
	DASIN	Generate radian from SIN value
	DASINP	Generate radian from SIN value (pulse type)
	DACOS	Generate radian from COS value
	DACOSP	Generate radian from COS value (pulse type)
	DATAN	Generate radian from TAN value
	DATANP	Generate radian from TAN value (pulse type)
	DRAD	Binary floating point degrees to radians conversion
	DRADP	Binary floating point degrees to radians conversion (pulse type)
	DDEG	Binary floating point radians to degrees conversion
	DDEGP	Binary floating point radians to degrees conversion (pulse type)
	DSINH	Floating point SINH operation
	DSINHP	Floating point SINH operation (pulse type)
	DCOSH	Floating point COSH operation
	DCOSHP	Floating point COSH operation (pulse type)
	DTANH	Floating point TANH operation
	DTANHP	Floating point TANH operation (pulse type)

Positioning Instruction

Instruction Type	Instruction	Function Description
Positioning Instruction	DABS	Absolute current value read
	ZRN	Setting of zero return speed (16 bit)
	DZRN	Setting of zero return speed (32 bit)
	PLSV	Variable speed pulse output (16 bit)
	DPLSV	Variable speed pulse output (32 bit)
	DRVI	Relative position control (16 bit)
	DDRVI	Relative position control (32 bit)
	DRVA	Absolute position control (16 bit)
	DDRVA	Absolute position control (32 bit)

Clock Control Instruction

Instruction Type	Instruction	Function Description

Clock Control Instruction	TCMP	Time compare
	TCMPP	Time compare (pulse type)
	TZCP	Time zone compare
	TZCPP	Time zone compare (pulse type)
	TADD	Time addition
	TADDP	Time addition (pulse type)
	TSUB	Time subtraction
	TSUBP	Time subtraction (pulse type)
	TRD	Time read
	TRDP	Time read (pulse type)
	TWR	Time write
	TWRP	Time write (pulse type)
	HOUR	16 bit stopwatch
	DHOUR	32 bit stopwatch

External Device Instruction

Instruction Type	Instruction	Function Description
External Device Instruction	GRY	16-bit Gray code conversion
	GRYP	16-bit Gray code conversion (pulse type)
	DGRY	32-bit Gray code conversion
	DGRYP	32-bit Gray code conversion (pulse type)
	GBIN	16-bit Gray code inverted conversion
	GBINP	16-bit Gray code inverted conversion (pulse type)
	DGBIN	32-bit Gray code inverted conversion
	DGBINP	32-bit Gray code inverted conversion (pulse type)
	RD3A	Read from analogue module
	RD3AP	Read from analogue module (pulse type)
	WR3A	Write to analogue module
	WR3AP	Write to analogue module (pulse type)

Comparison Instruction

Instruction Type	Instruction	Function Description
	LD=	Comparison of 16-bit data (==)
	LDD=	Comparison of 32-bit data (==)
	LD>	Comparison of 16-bit data (>)
	LDD>	Comparison of 32-bit data (>)
	LD<	Comparison of 16-bit data (<)
	LDD<	Comparison of 32-bit data (<)
	LD<>	Comparison of 16-bit data (<>)
	LDD<>	Comparison of 32-bit data (<>)
	LD<=	Comparison of 16-bit data (<=)
	LDD<=	Comparison of 32-bit data (<=)
	LD>=	Comparison of 16-bit data (>=)

Comparison Instruction	LDD>=	Comparison of 32-bit data (>=)
	AND=	Comparison of 16-bit data (==)
	ANDD=	Comparison of 32-bit data (==)
	AND>	Comparison of 16-bit data (>)
	ANDD>	Comparison of 32-bit data (>)
	AND<	Comparison of 16-bit data (<)
	ANDD<	Comparison of 32-bit data (<)
	AND<>	Comparison of 16-bit data (<>)
	ANDD<>	Comparison of 32-bit data (<>)
	AND<=	Comparison of 16-bit data (<=)
	ANDD<=	Comparison of 32-bit data (<=)
	AND>=	Comparison of 16-bit data (>=)
	ANDD>=	Comparison of 32-bit data (>=)
	OR=	Comparison of 16-bit data (==)
	ORD=	Comparison of 32-bit data (==)
	OR>	Comparison of 16-bit data (>)
	ORD>	Comparison of 32-bit data (>)
	OR<	Comparison of 16-bit data (<)
	ORD<	Comparison of 32-bit data (<)
	OR<>	Comparison of 16-bit data (<>)
	ORD<>	Comparison of 32-bit data (<>)
	OR<=	Comparison of 16-bit data (<=)
	ORD<=	Comparison of 32-bit data (<=)
	OR>=	Comparison of 16-bit data (>=)
	ORD>=	Comparison of 32-bit data (>=)

Detailed description of command

The detailed description of command is divided into three categories: basic sequential control command description, step sequential control command description, and function command (application command) description.

Basic Sequential Control Command

This section defines the types of progressive sequential commands and their functions.

Instructions ANB and ORB

Instruction Description

ANB instruction has not operands and the step number of program which ANB instruction shares is 1. The operand of ORB instruction can be X, Y, S, M, T, C, and the step number of program which ORB instruction shares is 1

Multi-branch loop uses ANB instruction When it series with the previous loop, Branch starts with LD, LDI, LDP, LDF, and use the ANB instruction to series with the branch which starts with LD, LDI, LDP, LDF instructions.

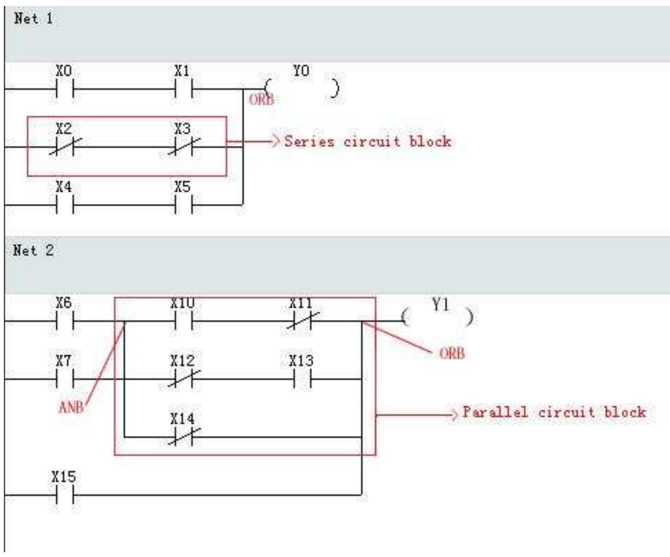
When serial circuit blocks of more than 2 contacts in series connect in parallel, every branch starts with LD and LDI instructions and ends with ORB instruction.

ANB and ORB instructions are not the instructions with the soft component.

The number of serial loop which ANB and ORB instructions uses is unlimited, but when used as approved, we must consider that the using the LD and LDI is in 8 times.

Programming Illustration:

Ladder mode:



Instruction List mode:

```
//
LD      X0
AND     X1
LDI     X2
ANI     X3
ORB
LD      X4
AND     X5
ORB
OUT     Y0
//
LD      X6
OR      X7
LD      X10
ANI     X11
LDI     X12
AND     X13
ORB
ORI     X14
ANB
OR      X15
OUT     Y1
```

ORB instruction is used in the end of each branch, not in the end of all branches, as command table above shown.

ORB and ANB instructions merely connect on the block. If not the block, not used. As shown, examples for series circuits block and parallel circuits block.

Instructions AND,ADNI,ANDP,ANDF

Instruction Description

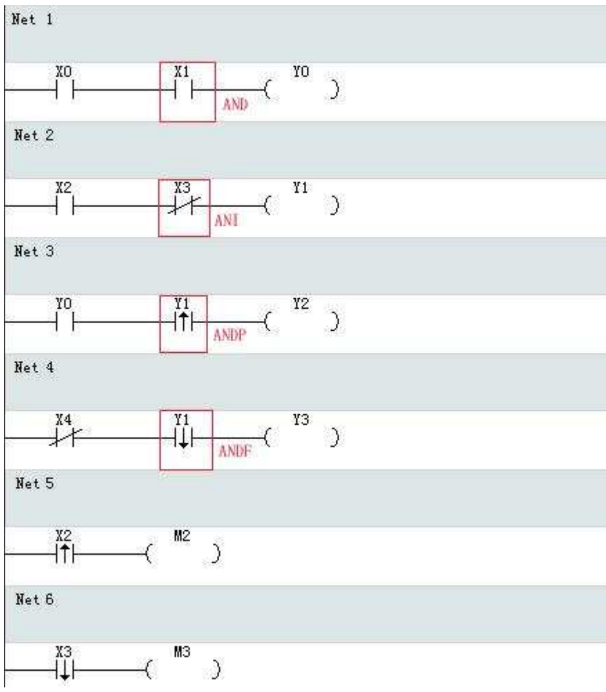
The steps of AND and ANI is 1, the steps of ANDP and ANDF is 2. The operands of these 4 instructions can be X, Y, S, M, T, C.

The instructions of AND, ADNI, ANDP and ANDF only contact one contact point. Two or more parallel circuits use ANB instruction when they are in series. The times in series is unlimited.

When ANDP and ANDF instructions is in the rising edge (when component change from the ON to OFF) and falling edge (when soft component change from the OFF to ON) contacts be connected for one cycle.

Programming Illustration:

Ladder mode:



Instruction List mode:

```
//
LD      X0
AND     X1
OUT     Y0
//
LD      X2
ANI     X3
OUT     Y1
//
LD      Y0
ANDP    Y1
OUT     Y2
//
LDI     X4
ANDF    Y1
OUT     Y3
//
LDP     X2
OUT     M2
//
LDF     X3
OUT     M3
```

- In the above example, X0, X3, Y1 are as contacts in series and conect with the front contact.

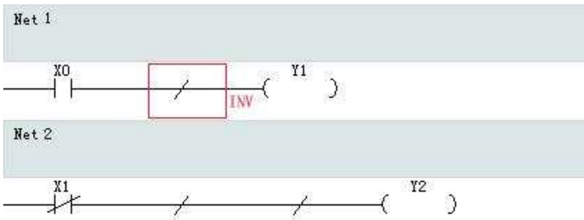
Instruction INV

Instruction Description

INV is the instruction which reverse the results before INV instruction and after LD, LDI, LDP, LDF instructions. And it has not operands. The instruction spend 1 process step.

Programming Illustration:

Ladder mode:



Instruction List mode:

```
//
LD      X0
INV
OUT     Y1
//
LDI     X1
INV
INV
OUT     Y2
```

Instructions LD,LDI,LDF,LDP,OUT

Instruction Description:

LD, LDI takes 1 process step. LDP、LDF takes 2 process steps. The operands of these 4 instructions can be X, Y, S, M, T, C.

The operand of OUT can be Y、S、T、M、or C .Soft component Y and the general M takes 1 process step. S and special auxiliary relay M take 2 process steps. Timer T takes 3 process steps. Counter takes 3-5 process steps.

LD、LDI、LDP、LDF makes the contact connected to bus bar. It is also used when Multiple branches with ANB, ORB.

LDP is connected for a cycle at the time of rising edge(Soft component changes from OFF to ON).LDF is connected for a cycle at the time of falling edge(Soft component changes from ON to OFF).

LD, LDI, LDP, LDF repeats less than 8 times. It means Maximum number of series and parallel connection is 8 when it is used with ANB、ORB behind.

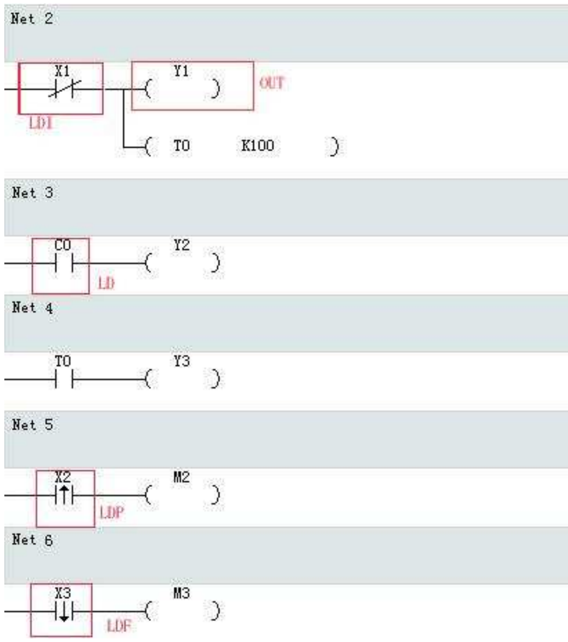
Soft component Y and the general M takes 1 process step. S and special auxiliary relay M take 2 process steps. Timer T takes 3 process steps. Counter C takes 3-5 process steps.

OUT drives Soft component coil except for Input Relay. OUT can be used continuously when used side by side.

When OUT drives counter and the front coil changes from ON to OFF or from OFF to ON ，the counter increase 1.

Programming Illustration:

Ladder mode:



Instruction List mode:

```
//
LD      X0
OUT     Y0
OUT     C0 K10
//
LD      X1
OUT     Y1
OUT     T0 K100
//
LD      C0
OUT     Y2
//
LD      T0
OUT     Y3
//
LDP     X2
OUT     M2
//
LDF     X3
OUT     M3
```

Use LD, LDI, LDP, LDF to connect with bus. Use OUT drives output coil.

When using OUT drives timing coil of timer or timing coil of counter, it is no need to set the time value and count value. It can be a constant K, or indirectly set by the register .

Instruction MC,MCR

Instruction Description:

The program step of MC instruction is 3 and the operands are Y, M (except for special M). The program step of MCR instruction is 2 and the operands are Y, M (except for special M).

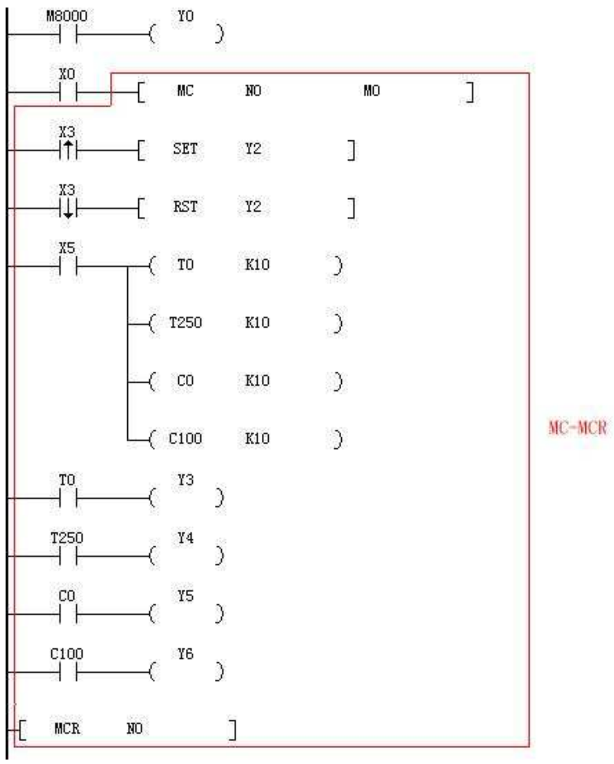
When previous contacts is connected, implement the MC and MCR instructions. when implementing the MC instruction, bus bar moves to MC contact, implement MCR instructions and return to bus bar.

When using MC instruction, the number K of the nested class increases by order, that is only the K0, to nesting K1. Instead, when using MCR instruction, it must return bus bar from large to small. Maximum nesting level is 7 (K6).

MC instruction can be used multiple times through different software components Y, M. If you use the same components, the same with the OUT instruction, there will be dual-coil output.

Programming Illustration:

Ladder mode:



Instruction List mode:

LD	M8000
OUT	Y0
LD	X0
MC	N0 M0
LDP	X3
SET	Y2
LDF	X3
RST	Y2
LD	X5
OUT	T0 K10
OUT	T250 K10
OUT	C0 K10
OUT	C100 K10
LD	T0
OUT	Y3
LD	T250
OUT	Y4
LD	C0
OUT	Y5
LD	C100
OUT	Y6
MCR	N0

This example only uses the MC, MCR instruction, the nested series is 1, 7 can be nested.

In this example, when X0 connected, MC and MCR instruction is implemented. When X0 is disconnected, two status as the following:

1) maintain the status : the value of the cumulative timer or counter value, use the SET / RST instruction to drive software components.

2) change into disconnected components : the value of the non-cumulative timer, use OUT instruction to drive software components.

Instruction MPS,MRD and MPP

Instruction Description

Instruction MPS,MRD and MPP have no operand, the share of program steps of all of these three instructions is one step.

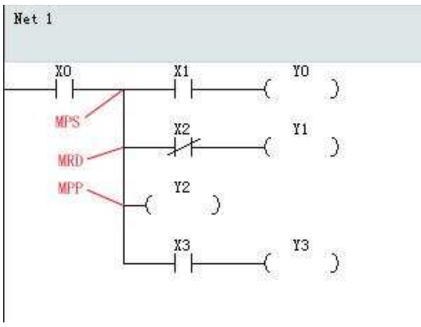
There are 11 stacks in the embedded PLC, that means the maximum depth of stacking is 11. Used once every instruction MPS, the current results are pressed into the first stack and stored, the results pressed previously moved to the next stack in turn. Instruction MPP read the first stack and delete it, the following unit move forward in turn at the same time.Instruction MRD read the first stack ,but it do not delete it. The other units remain unchanged. Using the three instructions can make multi-branch convenient.

When carrying out multi-branch program, instruction MPS saves the previous results, so that the branch behind can use instruction MRD and MPP to get the previous results in the stack and do the follow-up calculation. The last branch must use instruction MPP to make sure that the frequency of use of MPS and MPP is the same. Pay attention, after using MPP, you can't use MRD to read the result of calculation, that means MPP must be used in the last branch.

Instruction MRD can be used many times, there is no limit. The maximum number of continuous use of MPS is 11, but it can be used multiple times. Every instruction MPS has its corresponding instruction MPP, the number of MPP can't be more than that of MPS.

Programming Illustration:

Ladder Diagram (Illustration 1):

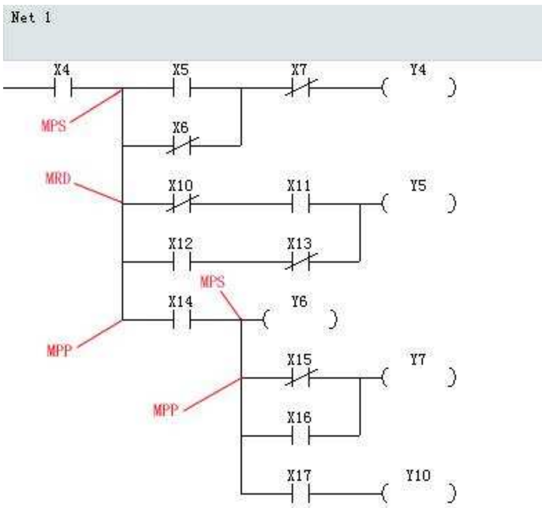


Instruction list mode(Example 1):

```
//
LD      X0
MPS
AND     X1
OUT     Y0
MRD
ANI     X2
OUT     Y1
MPP
OUT     Y2
AND     X3
OUT     Y3
```

Example 1 uses only one stack, uses an instruction of MPS to press stack, an instruction of MRD to read stack and an instruction of MPP to get out of the stack.

Ladder Diagram (Illustration 2):



Instruction list mode(Example 2):

```
//
LD      X4
MPS
LD      X5
ORI     X6
ANB
ANI     X7
OUT     Y4
MRD
LDI     X10
AND     X11
LD      X12
ANI     X13
ORB
ANB
OUT     Y5
MPP
AND     X14
MPS
OUT     Y6
LDI     X15
OR      X16
ANB
OUT     Y7
MPP
AND     X17
OUT     Y10
```

- Example 2 uses one level two stack, and uses it mixed with the struction of OR, ORB and ANB.

Instructions OR,ORI,ORP,ORF

Instruction Description

The steps of OR and ORI is 1, the steps of ORP and ORF is 2. The operands of these 4 instructions can be X, Y, S, M, T, C.

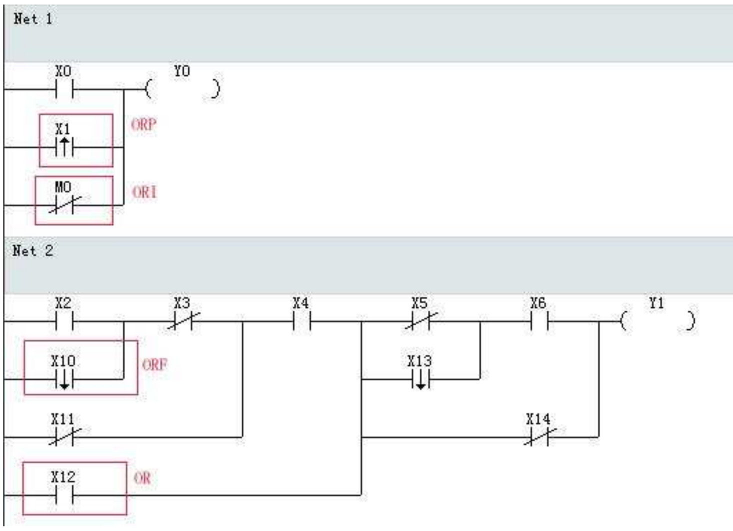
The instructions of OR, ORI, ORP and ORF only contact one points. Two or more series circuits use ORB instruction when they connect in parallel.

When ORP and ORF instructions is in the rising edge (when components change from the OFF to ON) and falling (when soft parts change from the ON to OFF), a cycle is connected.

When OR, ORI, ORP, ORF instructions and LD, LDI, LDP, LDF instructions are used together, the number of times in parallel is unlimited.

Programming Illustration:

Ladder mode:



Instruction List mode:

```
//
LD      X0
ORP     X1
ORI     M0
OUT     Y0
//
LD      X2
ORF     X10
ANI     X3
ORI     X11
AND     X4
OR      X12
LDI     X5
ORF     X13
AND     X6
ORI     X14
ANB
OUT     Y1
```

Use OR, ORI, ORP, ORF to connect with LD, LDI, LDP, LDF in parallel, the program has two parallel circuit blocks in series, so use the ANB instruction. The later chapters of the instructions introduce ANB instruction.

Instructions PLS,PLF

Instruction Description

The steps of program which PLS and PLF share are 1, and operands can be Y and M (except for special M).

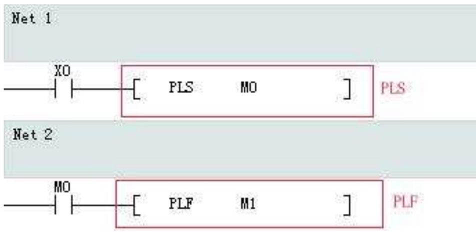
When using PLS instruction, driving software components in a scanning period when the coil changes from ON to OFF.

When using PLS instruction, driving software components in a scanning period when the coil changes from OFF to ON.

The components which have the function of latched generates run-time action when it runs the first time.

Programming Illustration:

Ladder mode:



Instruction List mode:

```
//
LD      X0
PLS     M0
//
LD      M0
PLF     M1
```

Instructions SET,RST

Instruction Description

The operands of SET instruction are Y, M, S; RST operands are X, Y, S, M, T, C, D, V, Z.

The steps of SET and RST instructions are determined by the following rules:

The procedure step of Soft component Y and the general M is 1,the program step of ?S and special auxiliary relay M, timer T, counter C is 2, the program step of data register D and variable access to sites both V and Z is 3.

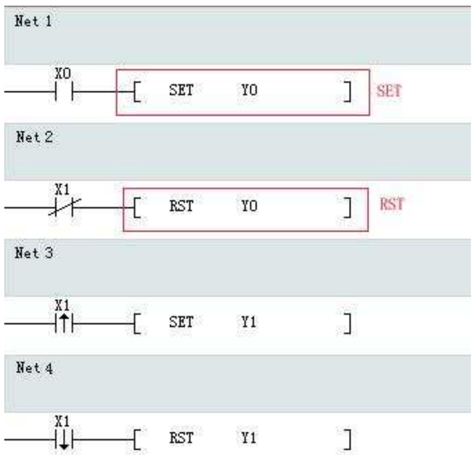
SET command set the soft component when the coil is connected, as long as the set position, unless reset the soft component with RST instruction, it will remain 1 as a state. Similarly, the RST instructions reset the soft component, and it will remain 0 as a state, unless using the SET command to set.

For the same soft component, SET and RST commands can be used multiple times and random order.

RST command can make data register D, index register V, Z, timer T, the counter C to reset and be zero, either maintained or non-maintained devices.

Programming Illustration:

Ladder mode:



Instruction List mode:

```
//
LD      X0
SET     Y0
//
LDI     X1
RST     Y0
//
LDP     X1
SET     Y1
//
LDF     X1
RST     Y1
```

Progressive Sequential Control Command

This section provides features and functions descriptions over the two progressive ladder commands: STL and RET.

STL and RET Instruction

Instruction Description

Mnemonic, Name	Function	Usable soft component	Program Step
STL	Initiation of Step Procedure	S	1
RET	End of Step Procedure	None	1

Step Control method (STL) divides controls into several operating procedures (S). Depending on the conditions in each procedure, status transitions will be carried out and complete the operation procedures progressively.

Step Control method's feature is that after taken into considerations for each control step and divided the complex procedure into successive steps, it greatly reduces the interdependence between each step and the complexity involved in programming.

Every movement executed in each status are programmed by other instructions in the ladder diagram.

STL is the initiation instruction for step procedures, and RET is the ending instruction for a step procedure. After the instruction is executed, it returns to the bus bar.

SET S[k]([k] is in decimal) is the only instruction for initiating STL transitions.

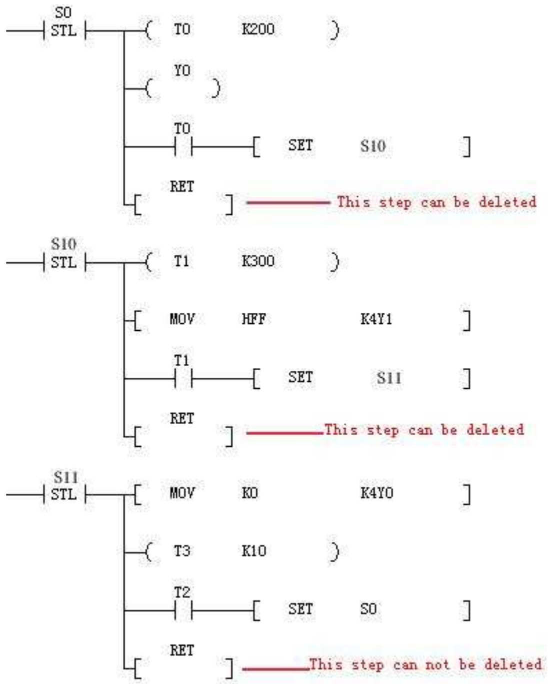
Rule: STL---RET instructions cannot be used in sub-programs.

When transition is happening from current status (S0) to next status (S1), the actions under the two scanning cycle conditions will both be executed; when the next scanning cycle is being executed, current status (S0) will be reset by the next status (S1), and the actions under the current status (S0) will not be executed. All OUT components' inputs will be interrupted.

Generally speaking, RET will be omitted between each step procedures. Therefore, it will seem a RET is shared by several STL. When STL is programmed and RET procedure is not, error message will appear.

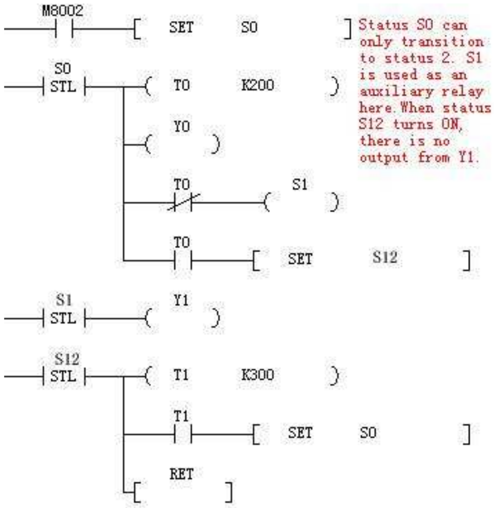
Programming Illustration:

Ladder Diagram (Illustration 1):



As illustrated above, RET is omitted between each step procedures. One RET is being shared by several STL. When STL is programmed and RET procedure is not, programming error message will appear.

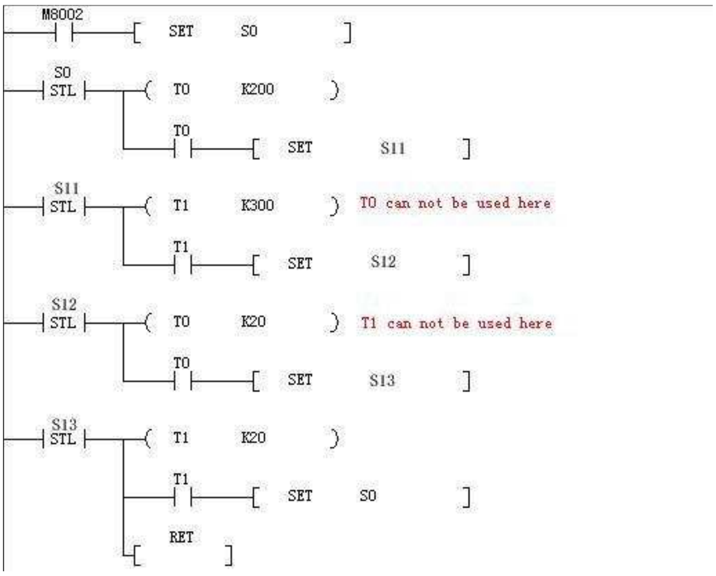
Ladder Diagram (Illustration 2):



As demonstrated in Illustration 2, only the SET instruction can be used in status transition and not the OUT instruction.

When using OUT S, S will be used as an assisting relay, instead of a status register.

Ladder Diagram (Illustration 3):



As demonstrated in Illustration 3, Time Relay T can be repeatedly used. However, the two neighboring statuses cannot use the same time relay repeatedly.

Commands & Functions

In this section the features and functions of application commands are described in detail; Commands that have the same functions (16-bit, 32-bit, progressive execution, and pulse commands types) will be described together.

ABSD instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ABSD	BIN addtion	16	No	ABSD S1 S2 D n	9
DABSD	operation	32	No		17

This instruction does a multi-section comparison, which is used for realizing cam control. The table and counter for comparison are all set in absolute mode. The instruction is implemented in the scanning main program, and the comparison result is affected by scan time delay. Where:

S1 is the starting component address of the comparison table.

S2 is the counter component serial number. When using 32 bit instruction, it could be used as a 32 bit counter.

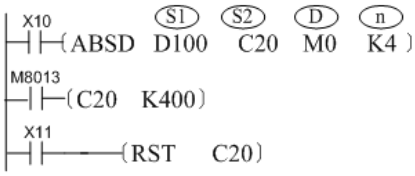
D is the starting address of the comparison result, occupying n several continuous bit variable units.

n is the number of multi-segment comparison data.

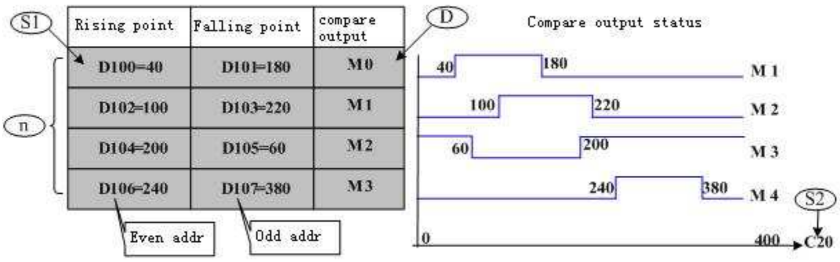
When using 32 bit instruction, S1 S2 D are all pointing to 32bit variable, and n is also calculated according to 32bit variable width.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1							✔	✔	✔	✔	✔	✔	✔		
S2												✔			
D				✔											
n	Constant,n＝1~64;														
When S1 operands are KnX、KnY、KnM、KnS , if it is 16bit instruction, K4 must be specified; if it is 32bit instruction, K8 must be specified and the component number of X,Y,M,S must be a multiple of 8. S1 operand can only specify C0 to C199 with 16bit instruction, and specify C200 to C254 with 32bit instruction.															

Programming example



If the relevant variables have been set as follows, when X10=ON, the implementation result is shown in the following figure.



Instruction for use:

Before ABSD instruction is implemented, all the variables in the form should be assigned with a MOV instruction.

- Even if the DABSD instruction is applied with high-speed instruction, the comparison result D is also affected by user program scan time delay. For the application with time response requirement, the HSZ high-speed comparison instruction is recommended.

ACOS instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DACOS	COS calculation for relevant radian	32	No	ACOS S D	9
DACOSP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

The instruction calculates the COS value for the relevant radian, where:

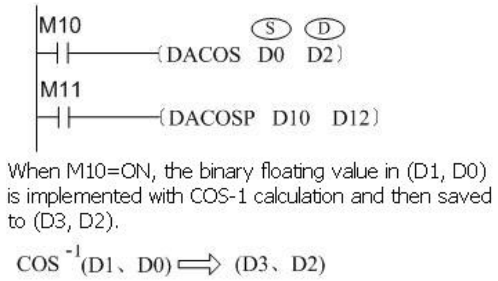
S is the binary floating variable for saving $\cos^{-1}$, which is to be calculated.

D is the storage unit (0~л)of the calculation result.

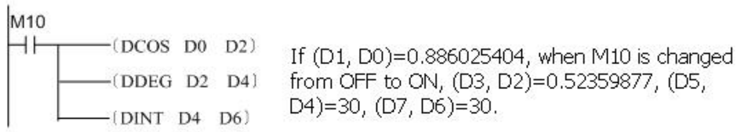
Note: If S is not within the range of -1.0 to 1.0, there will be a calculation error. The error code is K6706, which will be saved in D8067, and the error flag bit M8067 is set to ON.

Programming example

Example 1 for instruction:



Example 2 for instruction:



ADD instruction

Instruction Description:

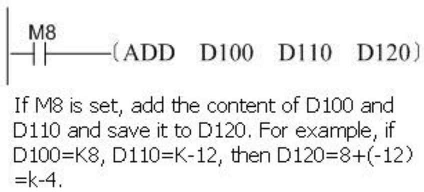
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ADD	BIN addtion operation	16	No	ADD S1 S2 D	7
ADDP		16	Yes		7
DADD		32	No		13
DADDP		32	Yes		13

This instruction is driven by contact with three operation variables. S1 and S2 are added in BIN algebra and saved in D. The involved variables are handled as a signed number, whose highest digit is a sign bit. 0 is positive number, and 1 is negative.

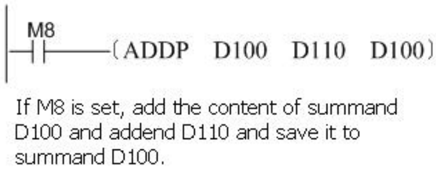
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

Programming example

Example 1 for instruction:



Example 2 for instruction:



Instruction for use

If the calculation result is 0, the 0 flag bit (M8020) will be reset.

- When the calculation result exceeds 32,767 (16bit calculation) or 2,147,483,647 (32bit calculation), the carry flag bit (M8021) will be reset.

- When the calculation result does not exceed -32,768 (16bit calculation) or -2,147,483,648 (32bit calculation), the carry flag bit (M8022) will be reset.

- When using 32bit calculation, the instruction variable address is a low 16bit address, and the adjoining address is a high 16bit address. It should be prevented from repeating or overwriting in the programming.

ALT instruction

Instruction Description

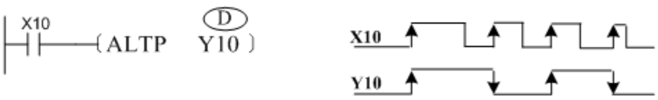
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ALT	Output alternatively	16	No	ATL D	3
ALTP		16	Yes		3

This instruction reverses D component state when the power flow is effective. D is bit variable component. Usually, the pulse operation type is preferred.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D		✔	✔	✔											

Programming example

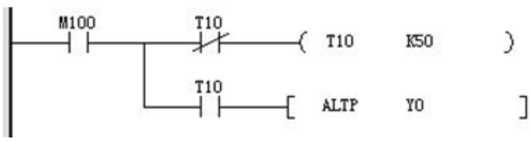
Example 1 for instruction:



The following instruction operation is the same:

Example 2 for instruction:

If the timer is introduced in the instruction power flow, it is easy to implement oscillator output (the function can also be implemented by using a special timer STMR instruction), which is shown in the following figure:



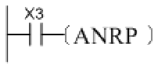
ANR instruction

Instruction Description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ANR	Signal alarm reset	16	No	ANR(without operand)	1

ANRP		16	Yes		1

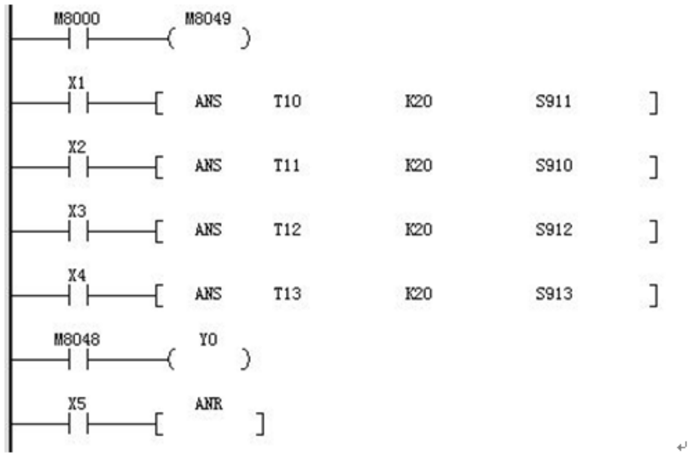
The ideal instruction is for a driver signal alarm. For example:



If X3 is connected, the alarm point with operation state in signal alarm S900~S999 is reset. If multiple alarm points are operating simultaneously, the alarm point with the lowest number is reset to ON.

If X3 is re-connected, the following number state is reset. Actually, ANRP instruction is preferred.

Programming example:



When M8049 is ON and any one bit in the range S900~S999 is ON, M8049 is set to ON, and Y0 signals the alarm.

If the program has S910, S911, S912, S913 all set to ON and X5 is switched from OFF to ON for the first time, S910 is reset.

When X5 is set to ON for the second time, S911 is simultaneously reset.

ANS instruction

Instruction Description

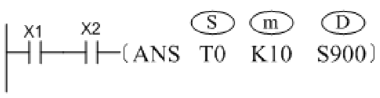
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ANS	Signal alarm setting	16	No	ANS S m D	7

The ideal instruction is for a driver signal alarm.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S											✔				
D				✔											
m	Constant，m=1~32767,（unit:100ms）.														

Where, the range of S is T0~T199, and the range of D is S900~S999.

Programming example:



If X1 and X2 are connected for more than 1 second, S900 is set. Following that, S900 stays in a state of operation, even if X1 or X2 is set to OFF (but T0 can be reset to 0). If X1 and X2 are connected for less than 1 second, X1 or X2 will set to OFF and the timer is reset.

If M8049 (signal alarm is available) is set to ON in advance, the lowest number with the ON state in signal alarm S900~S999 will be saved at D8049 (the lowest number with the ON state); when any signal in S900~S999 is ON then M8048 is set to ON (alarm operation).

ARWS instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ARWS	Directive switch	16	No	ARWS S D1 D2 n	7

The instruction specifies X as the edit key, and the Y port is a 4-digit, 7-fragment nixie tube, which is used as a simple interface for registering edited parameters, where:

S is the address where the specified key input begins, which occupies the following 4 bit units;

D1 is the variable that is displayed and modified, which is used to show only a variable with a 16bit width;

D2 is the starting address of the Y port of the nixie tube display driver, which occupies the following 8 Y ports.

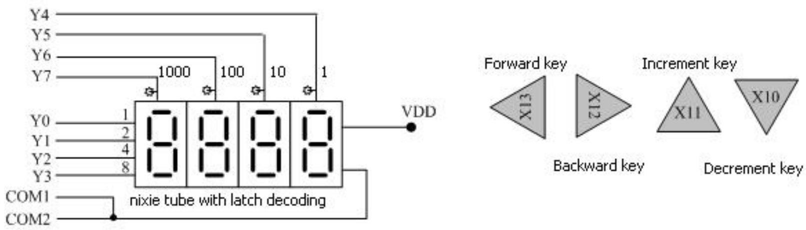
n is the value set for the logic signal, which refers to the n corresponding detailed description in the SEGL instructions above.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔	✔	✔	✔											
D1											✔	✔	✔	✔	✔
D2		✔													
n	Constant,n＝0~3														

Programming example

```
 X20        S    D1  D2  n
─┤├──(ARWS  X10  D0  Y0  K2)
```

The corresponding hardware wiring is shown in the following figure, in which PLC is the transistor output type:



Operation method:

(1)The nixie tube shows a figure value of D0. Press X10~X13 to modify the value, which should be within the 0~9999 range.

(2)When the X20 is ON, the cursor digit is shown as kilobits. Each time the backward key (X12) is pressed, the specified bit switches in the order of "thousand→hundred→ten→thousand"; when pressing the forward key (X13), the switch order reverses; and the digit cursor is indicated by the LED which is connected with the gating pulse signal (YO04 ~YOO7）.

(3)The cursor digit number switches in the order of 0 → 1→ 2→……8→9→0→1 when the increment key (X11) is pressed, when pressing the decrement key (X10), the number switches in the order of 0→9→8→7→…… 1→0→9,and the modified value becomes operative at once.

Instruction for use

When the scan time in the user program scan time is short, please use the constant scan mode instead, or scan in constant intervals using the interrupt timer.

ASC instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step

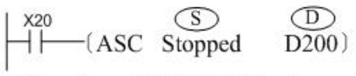
ASC	ASCII code conversion	16	No	ASC S D	11

S is the English letter inputted from computer, which is to be converted, and the max allowable length is 8 characters.

D is the starting component number used to storage ASCII code, which occupies successive 4 (M8161=0) or 8 (M8161=1) variables.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	When inputting instruction, it is inputted by a constant 8 characters in length.														
D											✔	✔	✔		

Programming example



The values of D200-D203 when X20=ON are shown in the left chart.

	High byte	Low byte
D200	54 (t)	53 (S)
D201	50 (p)	4F (o)
D202	45 (e)	50 (p)
D203	20	44 (d)

If the special register M8161 is set to ON, every ASCII character occupies one 16bit variable after conversion, which is shown in the following figure, and the higher byte of every variable is set to 0.

	High byte	Low byte
D200	00	53 (S)
D201	00	54 (t)
D202	00	4F (o)
D203	00	50 (p)
D204	00	50 (p)
D205	00	45 (e)
D206	00	44 (d)
D207	00	20

Attached: "ASCII code parallel fable"

Decimal digit	ASCII (Hex)
0	30
1	31
2	32
3	33
4	34
5	35
6	36
7	37
8	38
9	39

English letter	ASCII (Hex)	English letter	ASCII (Hex)
A	41	N	4E
B	42	O	4F
C	43	P	50
D	44	Q	51
E	45	R	52
F	46	S	53
G	47	T	54
H	48	U	55
I	49	V	56
J	4A	W	57
K	4B	X	58
L	4C	Y	59
M	4D	Z	5A

Code	ASCII (Hex)
STX	02
ETX	03

ASCI instruction

Instruction Description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ASCI	ASCII conversion	16	No	ASCI S D n	7
ASCIP		16	Yes		7

This instruction converts S value to ASCII, which is then saved in variable with start address D, where:

S is the variable address, which is to be converted, or a constant value.

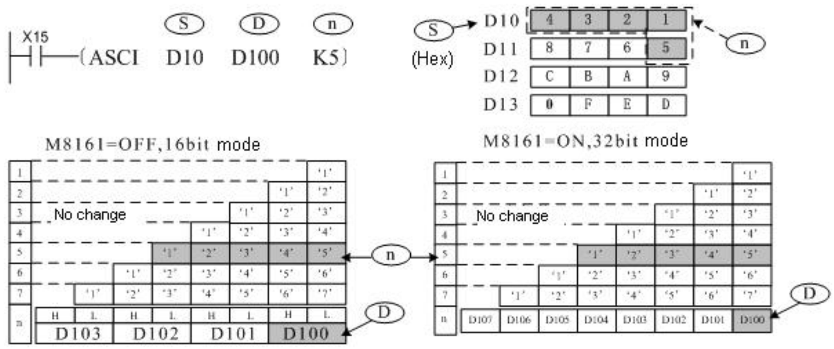
D is the start address for saving converted ASCII.

n is the converted character digit number.

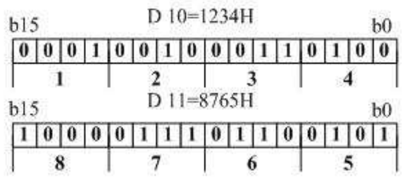
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔		
n	Constant, n=1~256														

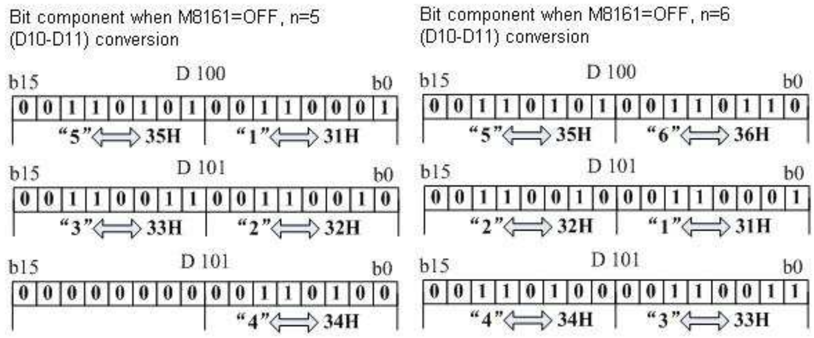
The ASCI value conversion complies with ASCII and HEX value parallel fable. For example: ASCII '0' according to HEX 'H30', ASCII 'F' according to HEX 'H46', and so on. For the contrast relationship of HEX and ASCII, please refer to the appendix following FNC76 (ASC) instruction.

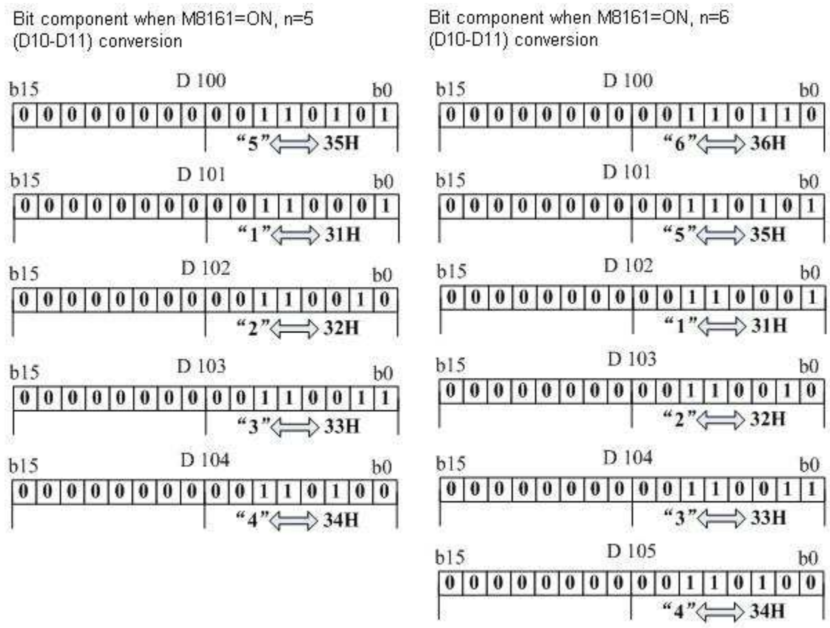
Programming example



The M8161 flag determines the width mode of the target variable for calculation result storage. When M8161=OFF, it is 16bit mode, which means the higher byte and lower byte are saved respectively. When M8161=ON, it is 8bit mode, which means that only the lower byte is used to save result and the actual variable range length is longer.







Note: RS/ HEX/ ASCI/ CCD instructions share the M8161 mode flag, which should noticed when programming.

ASIN instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DASIN	SIN calculation for relevant radian	32	No	ASIN S D	9
DASINP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

The instruction is to calculate the SIN value for the relevant radian, where:

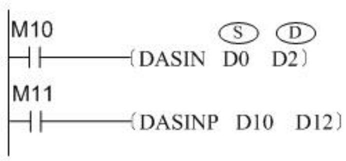
S is the binary floating variable for saving SIN^{-1}, which is to be calculated;

D is the storage unit for calculation of result (0~л).

Note: If the value of S is not within the range of -1.0~1.0, there will be a calculation error. The error code is K6706, which will be saved in D8067, and the error flag bit M8067 is set to ON.

Programming example

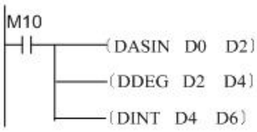
Example 1 for instruction:



When M10=ON, the binary floating value in (D1, D0) is implemented with SIN-1 calculation and then saved to (D3, D2).

SIN^{-1}(D1、D0) ⟹ (D3、D2)

Example 2 for instruction:



If (D1, D0)= 0.707106781, when M10 is changed from OFF to ON, (D3, D2)= 0.78539815, (D5, D4)= 45, (D7, D6)=45.

ATAN instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DATAN	TAN calculation for relevant radian	32	No	ATAN S D	9
DATANP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

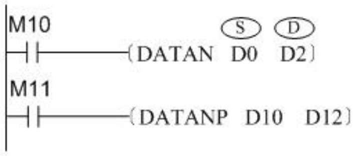
The instruction is to calculate the TAN-1 value for the relevant radian, where:

S is the binary floating variable for saving TAN^{-1}, which is to be calculated;

D is the storage unit for the calculation result(-л/2~л/2).

Programming example

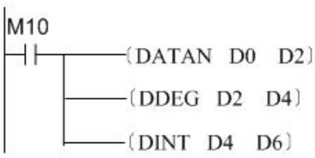
Example 1 for instruction:



When M10=ON, the binary floating value in (D1, D0) is implemented with TAN -1 calculation and then saved to (D3, D2).

TAN^{-1}(D1、D0) ⟹ (D3、D2)

Example 2 for instruction:



If (D1, D0) is 1.732050808, then the content of (D3, D2), (D5, D4) and (D7, D6) become 1.04719753, 60 and 60 when M10 is changed from OFF to ON.

BCD instruction

Instruction Description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
BCD	The instruction is to convert source (BIN) to target (BCD).	16	No	BCD S D	5
BCDP		16	Yes		5
DBCD		32	no		9
DBCDP		32	Yes		9

The instruction is driven by contact with two operation variables. S (BIN) value is converted in BIN and then saved to D. The instruction is usually used for data format processing before displaying.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S								✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

If conversion result exceeds 9999(16bit) or 99999999(32bit), there will be error.M8067, M8068 will be set to ON, and the error code will be saved in D8067.

Programming example:

The BIN value in D200 is converted to BCD value and the units of the result are saved to K1Y0 (four bit components Y0~Y3).
If D200=H000E (hex)=K14 (decimal), then Y0~Y3=0100(BIN) after conversion.
If D200= H0028 (hex)=K40 (decimal), then Y0~Y3=0000(BIN) after conversion.

BIN conversion

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
BIN	The instruction is to convert source (BCD) to target (BIN)	16	No	BIN S D	5
BINP		16	Yes		5
DBIN		32	no		9
DBINP		32	Yes		9

The instruction is driven by contact with two operation variables. S (BCD) value is converted into BIN and then saved to D. The instruction is usually used to convert the data, which is read from the external port, to BIN format, which can be directly applied for calculation.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S								✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

The available range of S(BCD) is 16bit:0~9999;32bit:0~99,999,999

If S data is not in BCD format, there will be a calculation error and M8067, M8068 will be reset.

Programming Illustration:

When M8 is set, the BCD value in K1Y0 is converted to BIN and saved to D200

BMOV instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
	Transmitting n				

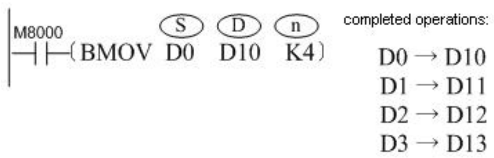
BMOV	data began with an original specified soft component to the address began with specified target soft component.	16	No	BMOV S D n	7
BMOVP		16	Yes		7

The instruction is driven by contact with three operation variables. n variables with a starting address specified by S are transmitted into units with a starting address specified by D.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔		
D								✔	✔	✔	✔	✔	✔		
n	Constant,n＝1~512														

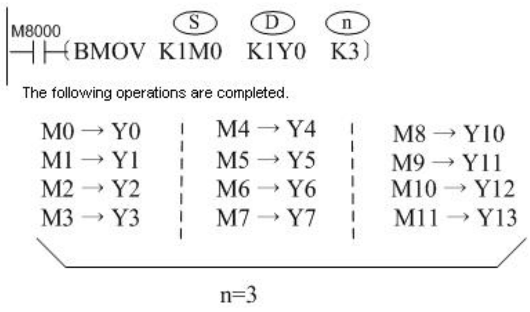
Where, n is within the range of 1~512.

When the special variable is M8024=1, the direct transmission is opposite, which means that n variables with a starting address specified by S are transmitted into n units with a starting address specified by D.



When operand is bit component, the digit number of S and D should be same.

Programming Illustration:



BON instruction

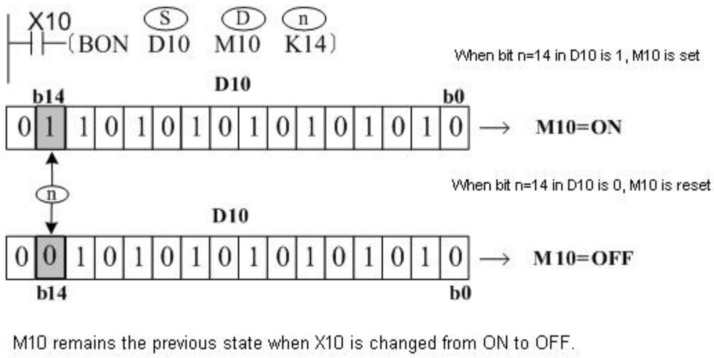
Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
BON	The instruction is to check whether the specific position is ON or OFF	16	No	BON S D n	7
BONP		16	Yes		7
DBON		32	no		13
DBONP		32	Yes		13

Judging the state of the No. n bit in S, and saving the result to D.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D		✔	✔	✔											
n	n=0~15（16bit）;n=0~31（32bit）														

Programming Illustration:



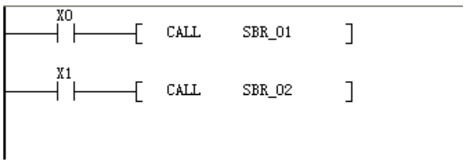
CALL instruction

Instruction description:

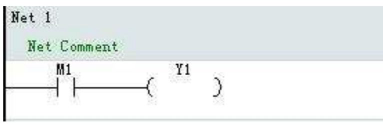
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
CALL	Subroutine call	16	No	CALL subroutine name	3
CALLP		16	Yes		3

Programming example:

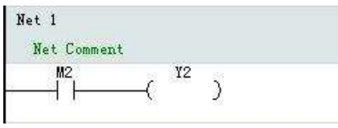
main program:



Subroutine SBR_01:



Subroutine SBR_02:



- According to the above example program, if X0 is ON, CALL instruction will be carried out to jump to subroutine SBR_01. If subroutine SBR_01 is completed, it will return to the main program to run the next instruction. Similarly, if X1 is ON, it will jump to subroutine SBR_02 to run, until program ends.

At most 4 nestings are allowed in a subroutine, that's to say, the maximum number of other subroutines permitted in any subroutine is 4.

CANRX instruction

Instruction description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
CANRX	CAN receive	16	No	CANRX S1 S2 D n	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		

S2					✔	✔							✔		
D													✔		
n					✔	✔							✔		

S1 is address 1:

S2 is address 0:

D is data buffer;

n is data length.

CANTX instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
CANTX	CAN transmit	16	No	CANTX S1 S2 D n	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
D													✔		
n					✔	✔							✔		

S1 is address 1;

S2 is address 0;

D is data buffer;

n is data length.

CCD instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
CCD	Checking code	16	No	CCD S D n	7
CCDP		16	Yes		7

This instruction checks and calculates n variables starting with S, and the addition result and logical exclusive-or one-by-one result are respectively saved to D and D+1. When used for communication, the instruction is implemented to string SumCheck for the correctness of data transmission.

S is the starting address and the following addresses are all used for saving variables, which are to be checked and calculated. D and D+1 are respectively used to save addition result and logical exclusive-or result, and n is the bit number occupied by variables for checking.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔		
D								✔	✔	✔	✔	✔	✔		
n	Constant,n=1~256														

Programming Illustration:

M8161 flag determines the variable width mode. When M8161=OFF, it is 16bit mode, which means that both higher bits and lower bits are involved in calculation. When M8161=ON, it is 8bit mode, which means that only lower bits are involved in calculation and higher bits are discarded, thus the actual variable range is extended, which is shown in the following figure:

"Accumulative summation" refers to the addition calculation result of specified n variables.

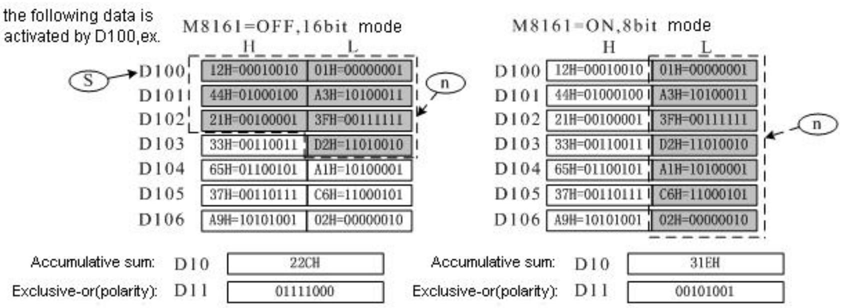
The "exclusive-or" logical calculation means:

1） The involved variables are converted to binary format.

2） Then it counts the number of variables with bit0=1. If it is even, the calculation result of bit0 is 0. If it is odd, the calculation result of bit0 is 1.

3） Then it counts the number of variables with bit1=1. If it is even, the calculation result of bit1 is 0; If it is odd, the calculation result of bit1 is 1.

4） In the same way, calculation is implemented from bit2 to bit7. After that, the binary HEX value converted from binary is the exclusive-or result (polarity value).



- RS/ HEX/ ASCI/ CCD instructions share the M8161 mode flag, which should be paid attention to when programming.

CJ,LBL instruction

Instruction description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
CJ	Conditional jump	16	No	CJ L0-L127	3
CJP		16	Yes		3
CJEND	Jump to program end	16	No	The single instruction without operands	3
CJPEND		16	Yes		3
LBL	Setting jump flag	16	No	LBL L0-L127	1

This instruction disables the sequence control program from CJ, CJP instruction to point (L). It can help to decrease circle time (scan period) and implement the program applying double coil.

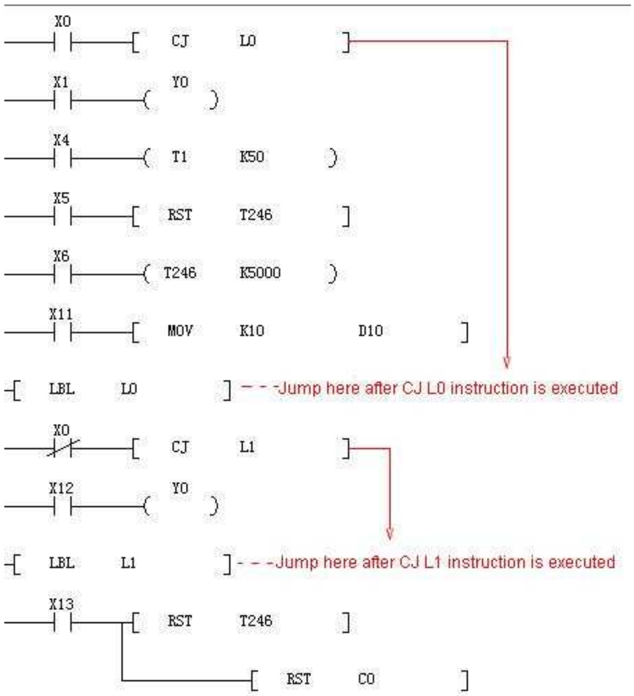
1 ）When power flow is effective, the program will automatically jump from the CJ (or CJP) instruction address to the address specified by L*** and go on running, and the skipped instructions will not be implemented.

2 ）When power flow is ineffective, the program will go on, and the CJ (or JCP) instruction will not be implemented.

If there is a TMR timer or counter in skipped instructions which has been activated, the operation should be:

Operation condition	CJ with jump	CJ without jump
T192~T199	Operating normally	Operating normally
Other timer	Stop timing	
C235~C255	Operating normally	
Other timer	Stop counting	

Programming Illustration:



- In the above example: If X0=ON and jump instruction is implemented, the coil operations in skipped instructions are listed as follows:
- Y,M,S hold the previous operation.
- If T is not activated before jumping, the timer will not operate even it is activated after jumping. If T is activated, it will keep running but contact will not operate. When X0 is OFF, contact operates immediately.
- If C is not activated before jumping, the counter will not operate even if it is activated after jumping. If it is activated, the timer interrupts. When X0 is OFF, the timer goes on counting.
- After jumping, the function instruction will not operate.

If the reset instruction of the timer and counter is out of the jump, the timer coil and jump counter coil reset is effective.

CML instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
CML	The instruction to transmit the data in reversion direction	16	No	CML S D	5
CMLP		16	Yes		5
DCML		32	No		13
DCMLP		32	Yes		13

The instruction is driven by contact with two operation variables. S (BIN) value is inverted bit by bit and then copied to D.

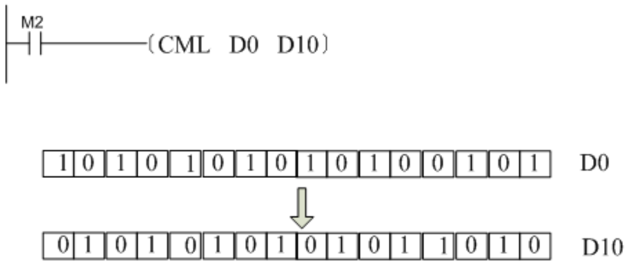
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

When D digit number is less than 16bit, it will result in inverting S and transmitting to D variable with low bit alignment.

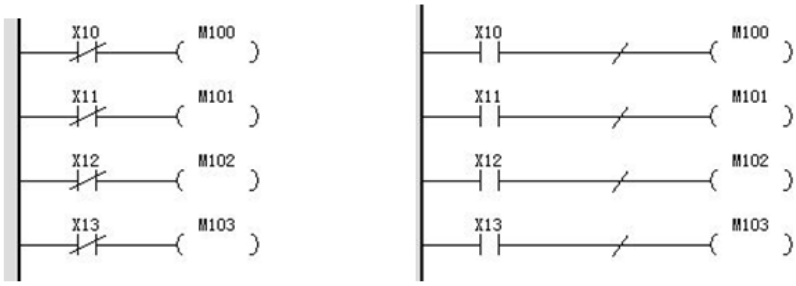
When it is 32 bit instruction (DCML), the corresponding S and D variable units in high address will be involved in calculation. Example： the calculation result for（DCML D1 D5）is:/D1→D5；/D2→D6

Programming Illustration:

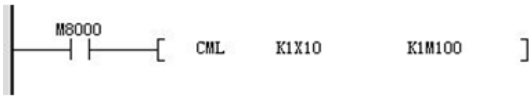
Example 1 for instruction:



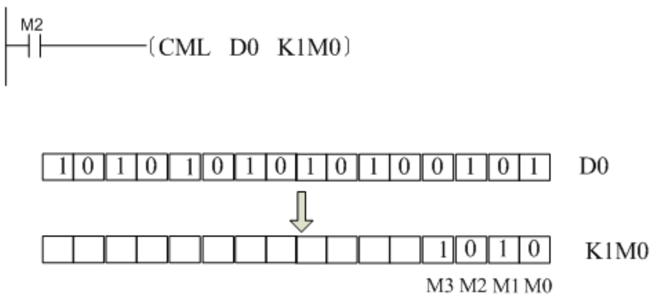
Example 2 for instruction:



The above-mentioned two programs can be implemented with the following CML instruction.



Example 3 for instruction:



CMP instruction

Instruction description:

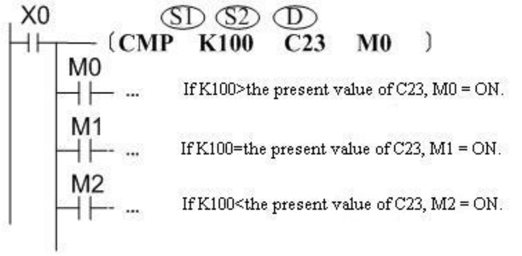
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
CMP	Comparison instruction	16	No	CMP S1 S2 D	7
CMPP		16	Yes		7
DCMP		32	No		13
DCMPP		32	Yes		13

This instruction compares two operational variables and outputs the comparison result to a specified bit variable. The operands are all algebra compared according to signed data.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D		✔	✔	✔											

D will occupy 3 bit variables in the continue address.

Programming Illustration:



One of M0~M2=ON if X0=ON.
CMP instruction will not be executed and M0~M2 will remain the state before X0= OFF when XO is changed from ON to OFF. RST or ZRST can be used to clear the comparison result of M0~M2.
M0~M2 can be connected in serial or parallel to obtain the result of ≧, ≦, ≠.

COSH instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DCOSH	COSH calculation for binary floating	32	No	COSH S D	9
DCOSHP		32	Yes		9

Operand	Bit component				Word component												
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z	
S					✔	✔	✔							✔			
D														✔			

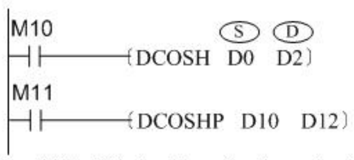
This instruction implements COSH calculation for binary floating. The calculation formula is cosh value=$(e^{s}+e^{-s})/2$,where:

S is the binary floating variables for saving COSH, which is to be calculated.

D is the storage unit for calculation result.

Programming Illustration:

Instruction example:



When M10=ON, the binary floating value in (D1, D0) is implemented with COSH calculation and then saved to (D3, D2).

DABS instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DABS	Read the current ABS value	32	No	DABS S D1 D2	13

The instruction is to read the motor absolute position (ABS) data from the servo driver via the high-speed input port.

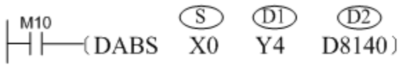
S is the input signal for reading the servo device, occupying the following three units.

D1 is the control signal transmitted to the servo device, occupying the following three units.

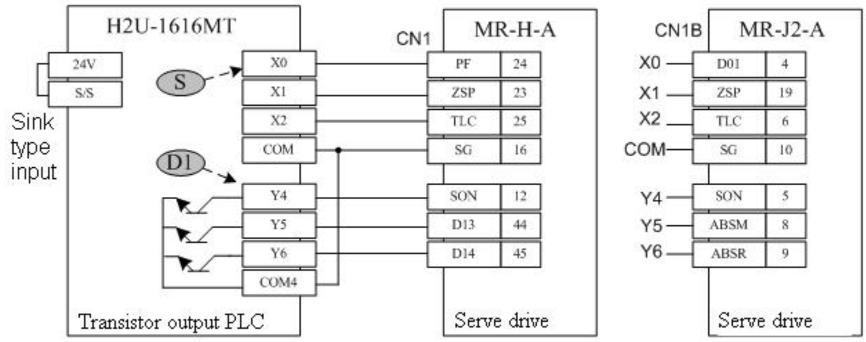
D2 is the storage unit for the data read from servo with a 32bit width, occupying D2+1, D2 unit that specifies D8140.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔	✔	✔	✔											
D1		✔	✔	✔											
D2								✔	✔	✔	✔	✔	✔	✔	✔

Programming Illustration:



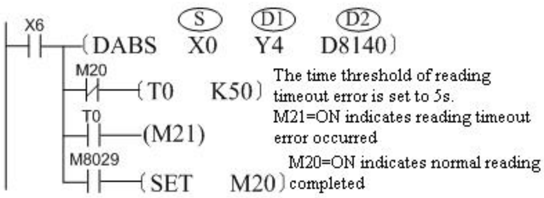
The corresponding wiring method is shown in the following figure, in which the servo driver is a Mitsubishi product equipped with an absolute position detection encoder servo motor.



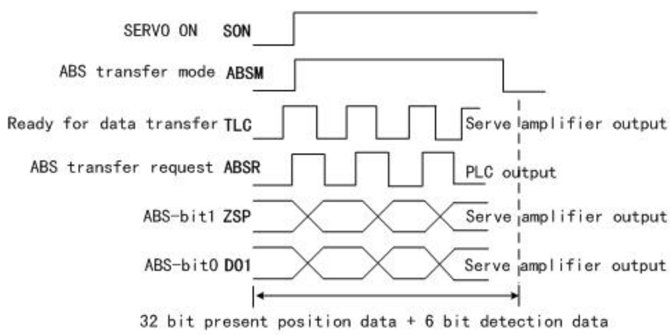
When the instruction driver M10 is set to ON, it begins to read. When this is completed, the M8029 flag is set to ON;

When the instruction implementation operation is in process and the driver flag is set to OFF, the read operation will be interrupted;

The programming example for reading ABS data is as follows: when the X6 terminal is closed, it begins to read. If it is not completed in 5s, the timeout flag M21 will be set. The code is listed as following:



The signal time sequence of the ABS read operation is shown in the following figure. When implementing an instruction, the PLC will automatically implement the access operation with servo driver.



DCOS calculation

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step

DCOS	Floating COS calculation	32	No	DCOS S D	9
DCOSP		32	Yes		9

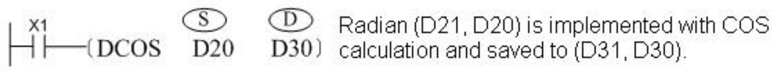
The instruction is to calculate the COS value for the specified angle (RAD, radian), in which the variables are in a binary floating format. Where:

S is the angle variable for the COS calculation, and the RAD unit is displayed in a binary floating point. With the available range of 0<=α<=2π;

D is the storage unit for the converted COS calculation results in binary floating format.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
D													✔		

Programming Illustration:



The calculated source data and COS results are all in binary floating format.

RAD(radian)value=angle×π/180°,for example, the radian corresponding to angle 360°=360°×π/180°=2π.

For the program instruction for the COS calculation of an angle, please refer to examples in the SIN instruction.

DEADD instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEADD	Binary floating addition	32	No	DEADD S1 S2 D	13
DEADDP		32	Yes		13

This instruction implements binary floating addition calculation.

S1 and S2 are respective binary floating addends.

D is the storage unit for the binary floating addition result.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
D													✔		

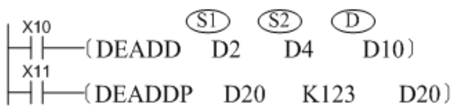
If the source operand of S1 or S2 is constant K or H, it will automatically be converted to binary floating value for addition calculation.

If the calculation result is 0, the 0 flag bit (M8020) will be reset.

If the calculation result absolute value is greater than the maximum displayable floating value, the carry flag (M8022) will be set.

If the calculation result absolute value is less than the minimum displayable floating value, the borrow flag (M8021) will be set.

Programming Illustration:



When X10=ON and binary floating variable (D3, D2) is added by binary floating variable (D5, D4), the result will be saved in (D11, D10).

When X11 is set from OFF to ON, the binary floating (D21, D20) value is added by 123. The constant K123 is automatically converted to binary floating value before calculation.

The storage unit for result could be the storage unit for addends, in which the pulse-type DEADD instruction is recommended, or the continue implementation instruction will be applied, in which the calculation will be implemented every time the program is scanned.

DEBCD instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEBCD	Converting binary floating to decimal floating	32	No	DEBCD S D	9
DEBCDP		32	Yes		9

This instruction converts binary floating to decimal floating.

S is binary floating variable.

D is the storage unit for converted decimal floating result.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
D													✔		

Programming Illustration:

The binary floating value in (D3,D2) is converted to decimal floating value and then saved to (D11,D10).
There are 23 bits real number, 8 bits exponent, and1 bit signal in binary floating [D3,D2], which will be converted to decimal floating [D11,D10], and it could be expressed with science formula of D2×10D3.

The floating data calculation in PLC is all in binary format, and it is converted to decimal for ease of monitoring.

DEBIN instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEBIN	Converting decimal floating to binary floating	32	No	DEBIN S D	9
DEBINP		32	Yes		9

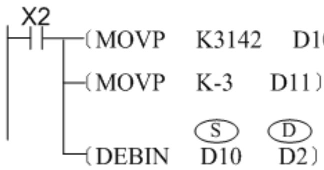
This instruction converts decimal floating to binary floating. Where:

S is decimal floating variable.

D is the storage unit for converted binary floating result.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
D													✔		

Programming Illustration:



The decimal floating 3.142, which is saved in D11,D10, is converted to binary floating and then saved in (D3,D2).

DEC instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEC	1 subtracted from BIN	16	No	DEC D	3
DECP		16	Yes		3
DDEC		32	No		5
DDECP		32	Yes		5

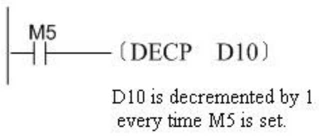
Everytime instruction be executed,subtracted 1 from D.

When 16bit operation, -32,768 subtracts 1 to 32,767;32bit operation, -2,147,483,648 subtracts 1 to 2,147,483,647. The instruction don't refresh in sign 0,carry and borrow.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D								✔	✔	✔	✔	✔	✔	✔	✔

When executed 32bit operation, D variable address is low 16bit address in instruction,border upon high coding address unit is high 16bit,be careful of repeat and cover in fault when program.

Programming Illustration:



DECMP instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DECMP	Binary floating comparison	32	No	DECMP S1 S2 D	13
DECMPP		32	Yes		13

This instruction compares two floating variables and outputs the comparison result to three variables starting with D. Where:

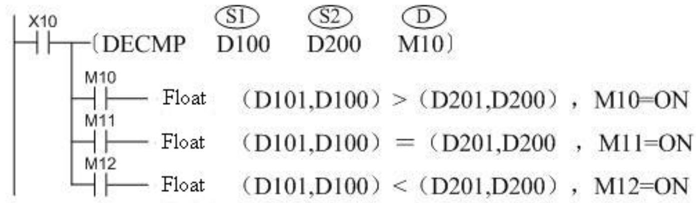
S1 is the binary floating value 1 for comparison.

S2 is the binary floating value 2 for comparison.

D is the storage unit for comparison result, occupying three variable units.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
D		✔	✔	✔											

Programming Illustration:



One of M10~M12 will be turned ON if X10=ON.
DECP instruction will not be executed and M10~M12 will remain the state before X0= OFF when X10 is changed from ON to OFF. RST or ZRST can be used to clear the comparison result of M10~M12.
M10~M12 can be connected in serial or parallel to obtain the result of ≧, ≦, ≠
.

If (S1) or (S2) is K, H constant, they will be automatically converted to floating for calculation.

DECO instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DECO	Code translation (to convert any one digit in data to one point with ON)	16	No	DECO (S) (D) (n)	7
DECOP		16	Yes		7

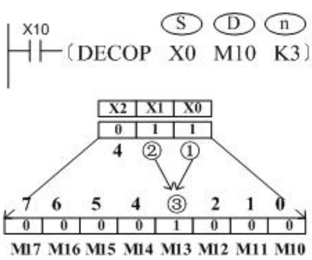
It calculates the value of (S) last (2^(n)) digit and takes it as the bit pointer. It sets the digit corresponding (D) to 1, and the other digits to 0.

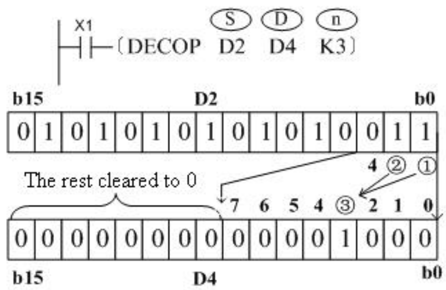
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(S)	✔	✔	✔	✔	✔						✔	✔	✔	✔	✔
(D)		✔	✔	✔							✔	✔	✔		
(n)	Constant, n=1~8. If n=0, the instruction will not be implemented. If n ≠0, there will be an implementation error.														

- The low n bit(s) (n≤4) of the source address is translated to target address. If n≤3, the higher bits of the target address is set to 0;
- If n=0, the instruction is not implemented. If n is not within the range of 0 to 8, there will be a calculation error.
- When n=8, if the code translation instruction is bit soft component, the point number is 256.
- When driver output is OFF, the instruction is not implemented and the code translation output in operation will be implemented.

The instruction usually uses pulse operation type instruction.

Programming Illustration:





DEDIV instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEDIV	Binary floating division	32	No	DEDIV S1 S2 D	13
DEDIVP		32	Yes		13

This instruction is to implement binary floating division calculation, where,

S1 and S2 represent binary floating dividend and divisor respectively;

D is the starting address of the binary floating division result storage unit.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
D													✔		

If the source operand of S1 or S2 are constant K or H, it will be automatically converted to a binary floating value for division calculation;

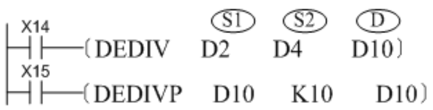
If the calculation result is 0, the 0 flag bit (M8020) will be reset.

If the calculation result absolute value is greater than the maximum displayable floating value, the carry flag (M8022) will be set.

If the calculation result absolute value is less than the minimum displayable floating value, the borrow flag (M8021) will be set.

The divisor should not be 0, or there will be an error and M8067, M8068 will be set to ON.

Programming Illustration:



When X14=ON and the binary floating variable (D3,D2) are divided by the binary floating variable (D5,D4), the result will be saved in (D11,D10).

When X15 is set from OFF to ON, the binary floating (D11,D10) is divided by 10 and then the result is saved back to (D11,D10).The constant K10 is automatically converted to a binary floating value before calculation.

The storage unit for the result could be the storage unit for the dividend or divisor, in which the pulse-type DEDIVP instruction is recommended, or the continue implementation instruction will be applied, in which the calculation will be implemented every time when the program is scanned.

DEG instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DDEG	Binary floating radian calculation for angle	32	No	DEG S D	9
DDEGP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

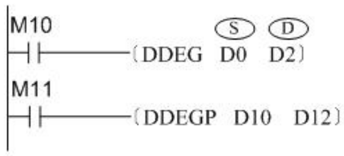
This instruction is the binary floating radian calculation for angle. The calculation formula is [Angle unit= radian unit×л/180],where:

S is the binary floating radian variable for saving angle, which is to be calculated ;

D is the storage unit for the calculation result.

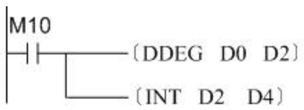
Programming Illustration:

Instruction Example:



When M10=ON, the binary floating value in (D1, D0) is implemented with radian-to-angle calculation and then saved to (D3, D2).

Instruction Example:



If (D1, D0)=3.1415926, (D3, D2)=180 when M10 is changed from OFF to ON. (D5, D4)=180 after floating-point number to integer calculation

DEMUL instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEMUL	Binary floating-point multiplication	32	No	DEMUL S1 S2 D	13
DEMULP		32	Yes		13

The instruction performs multiplication operation based on the binary system, where：

S1 and S2 represents the multiplicand and the multiplier in the binary system.

D is the product storage unit of the binary floating-point multiplication.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
D													✔		

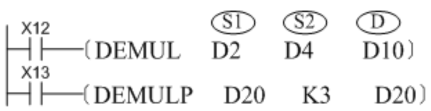
Should the source operand of S1 or S2 be constant K or H, it will automatically convert the constant to a binary floating-point value to further perform multiplication operation.

If the calculation result is 0, the 0 flag bit (M8020) will be reset.

If the calculation result absolute value is greater than the maximum displayable floating value, the carry flag (M8022) will be set.

If the calculation result absolute value is less than the minimum displayable floating value, the borrow flag (M8021) will be set.

Programming Example:



When X12 = ON, after the binary floating-point (D3, D2) multiplies the other binary floating-point (D5, D4), the product will be stored in (D11, D10).

When X13 turns from OFF to ON, the binary floating-point (D21, D20) value will be multiplied by 3 (three) and saved back in (D21, D20) The constant K3 has already been automatically converted to a binary floating-point value prior to the calculation.

The storing unit for the multiplication product can be treated as one unit with the multiplicand and the multiplier. Please use the pulse execution instruction DEMULP under this circumstance. Otherwise, if selected the progressive execution instruction, the multiplication operation will be carried out again every time when the program is scanned.

DESQR instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DESQR	Binary floating-point square root	32	No	DESQR S D	9
DESQRP		32	Yes		9

The command performs the square root calculation of the binary floating-points, where:

S is the binary floating-point variable that is to be square rooted.

D is the product storage unit of the binary floating-point square root.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔							✔		
D													✔		

Should the operand S is the constant K or H, it will be automatically converted to a binary floating-point value and square rooted; if the result of calculation is zero, it will be flagged and positioned at M8020.

S will only be effective when the value is positive. There must be errors in the calculation if it appears as negative. In this case, M8067, M8068 will be positioned as ON.

Programming Example:

The binary floating radication result is saved to（D11， D10）.
The binary floating number K6789 is implemented with radication calculation and then the result is saved to (D21, D20), where the constant K6789 is automatically converted to binary floating data before implementation;

DESUB instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Applicable operand soft components	Step
DESUB	Binary floating-point subtraction	32	No	DESUB S1 S2 D	13
DESUBP		32	Yes		13

The instruction performs subtraction operation based on the binary floating-point system, where:

S1 and S2 represents the minuend and the subtrahend in the binary floating-point system.

D is the difference storage unit of the binary floating-point subtraction.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
D													✔		

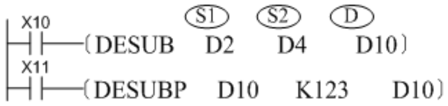
Should the source operand of S1 or S2 be constant K or H, it will automatically convert the constant to a binary floating-point value to further perform subtraction operation;

If the calculation result is 0, the 0 flag bit (M8020) will be reset.

If the calculation result absolute value is greater than the maximum displayable floating value, the carry flag (M8022) will be set.

If the calculation result absolute value is less than the minimum displayable floating value, the borrow flag (M8021) will be set.

Programming Example:



When X10 = ON, after the binary floating-point (D3, D2) subtracts the other binary floating-point (D5, D4), the difference result will be stored in (D11, D10).

When X11 turns from OFF to ON, the value of the binary floating-point requires to subtract 123.The constant K123 is automatically converted to binary floating value before calculation.

The storing unit for the subtraction difference can be seemed as same one unit with the subtrahend and minuend. Please use the pulse execution instruction DESUBP under this circumstance. Otherwise, if selected the progressive execution instruction, the subtraction operation will be carried out again every time when the program is scanned.

DEZCP instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEZCP	Binary floating-point zone comparison	32	No	DEZCP S1 S2 S D	17
DEZCPP		32	Yes		17

The instruction compares the inter-zoning variables of binary floating-points, and then exports the result to the three (3) initiative variables, where:

S1 represents the inter-zoning minimum of the binary floating-point variables.

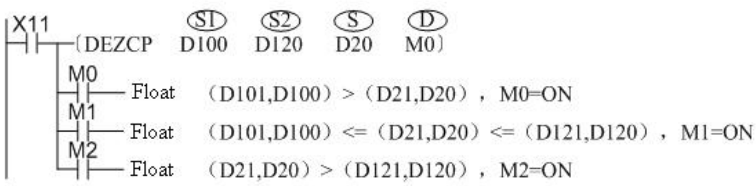
S2 represents the inter-zoning maximum of the binary floating-point variables.

S represents the binary floating-point variable that is to be compared.

D is the storage unit for comparison result, occupying three variable units.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
S					✔	✔							✔		
D		✔	✔	✔											

Programming Example:



DHSCR instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
	Comparison Reset (Every time when the system is counting, after it has compared				

DHSCR	the counted value and the assigned value, the system immediately resets the external output (Y))	32	No	DHSCR S1 S2 D	13

When S2 counter's current value equals the assigned value of S1, it resets to D, where:

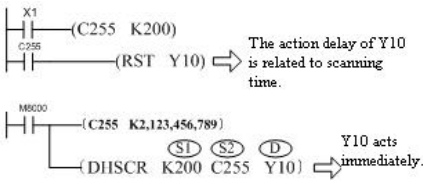
S1 is the designated comparison value. The value's scope (in bit) depends on the bit value of the S2 counter.

S2 variable must correspond to the high-speed counter C235~255. Because the counters engaged are all 32-bit counters, the 32-bit instruction, DHSCR, must be used.

D represents the storage unit of the comparison result: when the resulting port range is between Y0~Y17, results will be immediately exported; when the port is after Y20, results will only be exported after the user program has completed the scanning; when M and S becomes variables, the system will immediately refresh the storage unit.

Operand	**Bit component**				**Word component**										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2												✔			
D		✔	✔	✔											

Programming Example:



Instruction Instruction:

The operating principle of HSCR instruction is similar to that of HSCS instruction. The only difference is that the comparison exporting operation of HSCR is opposite of the HSCS instruction, which means that assigned export reset will only initiate after the counter has reached the same value with the designated value. Please refer to the instruction in the HSCS section.

DHSCS instruction

Instruction description:

Name	**Function**	**Bits(bits)**	**Pulse type**	**Instruction format**	**Step**
DHSCS	Comparison Reset (Every time when the system is counting, after it has compared the counted value and the assigned value, the system immediately resets the external output (Y))	32	No	DHSCS S1 S1 D	13

When S2 counter's current value equals the assigned value of S1, it resets to D, where:

S1 is the designated comparison value. The value's scope (in bit) depends on the bit value of the S2 counter.

S2 variable must correspond to the high-speed counter C235~255. Because the counters engaged are all 32-bit counters, the 32-bit instruction, DHSCS, must be used.

D represents the storage unit of the comparison result: when the resulting port range is between Y0~Y17, results will be immediately exported; when the port is after Y20, results will only be exported after the user program has completed the scanning; when M and S becomes variables, the system will immediately refresh the storage unit.

When (D) is between I010~I060, the subprogram for interrupting 0~5 in the high-speed counter needs to be initiated. It is certain that the corresponding interrupting subprogram, the initiation of relevant interrupting permissible signal, and the overall interrupting permissible signal must be properly programmed in order to intercept the counter when necessary. M8059 that is positioned as ON prohibits all intercepting procedures over high-speed counters.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2												✔			
D2		✔	✔	✔											

Differences between Y outputs under general and DHSCS instructions: for instance

1. When the present value of C255 is changing from 99 to 100, the C255 contact point will immediately become conductive. However, when the system is running the procedure to OUT Y10, because Y10 is still under the scanning cycle influence, it will only be exported after the process has reached END.

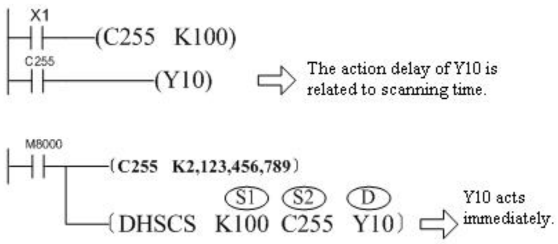
2. When the present value of C255 is changing from 99 to 100 and 101 to 100, the Y10 output under the DHSCS instruction is exported immediately to the external output port in an interceptive manner. It has no relevance to the PLC scanning cycle.However, the output will still be delayed due to the influences of the output module relay or transistor outputs.

High-speed interruption indicator and setup:

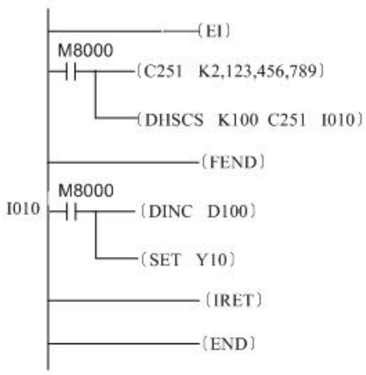
Operand	Interruption Prohibiting Instruction
I010	M8059
I020	
I030	
I040	
I050	
I060	

Programming Example

Programming Illustration 1:



Programming Illustration 2:



The D operand range for DHSCS instruction can also be specified I0□0, □=1~6. Interrupt occurs when counter reaches the setting value.
All high speed counter interrupts are disabled if M8059=ON.
Note that ON type difference between I010 and output point Y, M, S used by D device in this case:
Y output point: If the present value of C251 is changed from 99 to 100 or 101 to 100, Y will turn ON immediately and remain ON all the time. After that, even if the compare result of C251and K100 becomes unequal, Y will remain ON unless other reset instruction is executed.

Instruction for use:

When using the HSCS instruction please ensure the counters have already been activated (see instruction example 1). Otherwise the counter's value will not change.

- The counter uses an interceptive method to influence the counter's input signals and perform real-time comparison. Should the comparison satisfies the matching relation, comparison output will be reset immediately. Take instruction example 1 for instance, when the present value of C255 changes from 99 to 100, or from 101 to 100, Y10 will be reset and remain in the reset state. Even if the comparison results of C255 and K100 are not equal, Y10 will still remain in ON status until other reset instruction has been introduced.

- System instruction's comparison output is determined based on the pulse input comparison result. If there was no pulse input, even if editing instructions such as DMOV or DADD is used to edit the contents in the high-speed counter C235~C255, comparison output will still bring no difference. Also, using any instruction to initiate power flow will not be able to change the result as well.

- If Y port is used for instruction output, it must be within the range from Y0 to Y17. Only then the immediate response can be guaranteed; Initiate the HSCS instruction multiple times or initiate the instruction with the HSCR and HSZ commends. This way the top 2 digits of the target output Y can be treated as a soft component of the same serial number. Example: Y000~Y007 when using Y000; Y010~Y017 when using Y010;

- When the HSCS instruction's output target is to interrupt I010~I060, every interrupting signal can only be used once and cannot be repeated.

- Like other general instructions, HSCS, HSCR, and HSZ instructions can be used multiple times. However, six is the limit number to execute these instructions simultaneously.

DHSZ instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DHSZ	inter-zoning compare	32	No	DHSZ S1 S2 S D	17

According to the present value of the counter S, comparisons will be conducted with the designated S1 and S2 inter-zoning values. Comparison results will be immediately exported to the initial three units starting from the D address, where:

S1 represents the inter-zoning minimum of the designated comparison zones. The value's width (in bit) is determined based on the bits of the S counter. The value must be no greater than S2. Therefore, S1=S2;

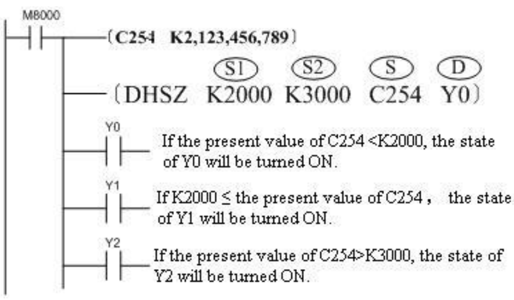
S2 represents the inter-zoning maximum of the designated comparison zones. The value's width (in bit) is determined based on the bits of the S counter. The value must be no less than S1. Therefore, S1=S2;

S variable must correspond to the high-speed counter C235~255. Because the counters engaged are all 32-bit counters, the 32-bit instruction, DHSZ, must be used.

D represents the storage unit of the comparison result, which uses the first three units with sequential addresses starting from D: when the resulting port range is between Y0~Y17, results will be immediately exported; when the port is after Y20, results will only be exported after the user program has completed the scanning; when M and S becomes variables, the system will immediately refresh the storage unit.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S												✔			
D		✔	✔	✔											

Programming Example:



Instruction for use:

The operating principle of this instruction is the same as the HSCS and HSCR instructions. There differences are that it uses two comparative values, and the comparison output uses three sequential address units. Therefore, some operating note can be referred to the operation instruction in the HSCR section; HSZ instruction also uses interruptive method in its operation. The comparison process and the regeneration of corresponding output will only proceed when there are counting pulses happening at the counter's corresponding input end;

DIV instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DIV	BIN Division Operation	16	No	DIV S1 S2 D	7
DIVP		16	Yes		7
DDIV		32	No		13
DDIVP		32	Yes		13

The instruction requires contact points activation and three operating variables. Values of the dividend S1 and divisor S2 will first multiply with the BIN algebra, and then the result is saved in D. All variables involved in the operation are processed according to the symbol number， and the highest is the symbol bit. 0 represents as positive, and 1 as negative.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S								✔	✔	✔	✔	✔	✔		

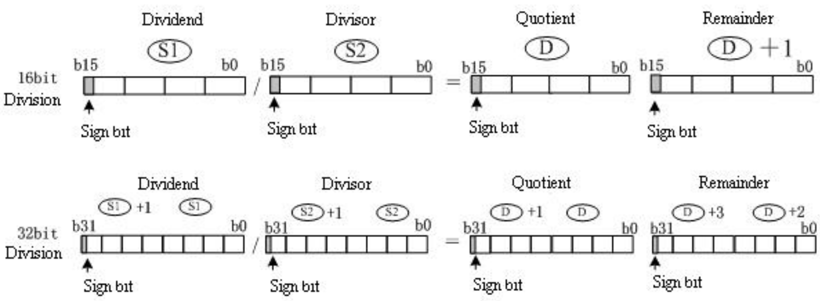
The V and Z components in the chart can only be used in 16-bit operation.

While performing 32-bit operation, the S1 and S2 variable addresses in the instruction are low 16-bit addresses. The neighboring high serial address units are high 16-bit and are used to prevent duplicates or erroneous re-writes during programming. The calculated quotient will be saved in the indicated D and D+ 1 unit. The remainder will be saved in D+2 and D+3 address units.

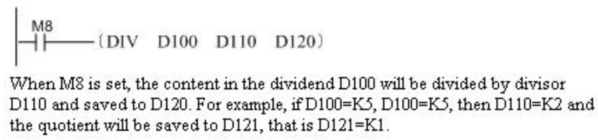
If the divisor S2 equals two, erroneous calculation will occur;

If bit components (KnX/KnY/KnM/KnS) are assigned as D, no remainder will be obtained;

If the dividend is negative, remainder will be negative as well.



Programming Example



DRVA instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DRVA	Absolute Positioning	16	No	DRVA S1 S2 D1	9
DDRVA		32	No		17

Based on the designated port, frequency, and the direction of the output pulses, the instruction allows the server to commence machinery movement to an appointed destination. Only the PLC with the transistor output can execute the instruction, where:

S1 is the designated target position (absolute position). When commend is in 16-bit, the range is from -32,768 to 32,767; when it is in 32-

bit, the range is from -2,147,483,648 to 2,147,483,647.

If D1 = [Y000], the corresponding [D8141 (high byte), D8140 (low byte)] (in 32-bit) will become the absolute position.

If D1 = [Y001], the corresponding [D8143 (high byte), D8142 (low byte)] (in 32-bit) will become the absolute position.

If D1= [Y002], the corresponding [D8151 (high byte), D8150 (low byte)] (in 32-bit) will become the absolute position.

If D1 = [Y003], the corresponding [D8153 (high byte), D8152 (low byte)] (in 32-bit) will become the absolute position.

If D1 = [Y004], the corresponding [D8155 (high byte), D8154 (low byte)] (in 32-bit) will become the absolute position. The negative symbol represents the opposite direction. When in reverse, the value of the current value register will reduce.

S2 represents the designated output pulse frequency, and it ranges from 10 to 32,767Hz (in 16-bit instruction); or from 10 to 100,000Hz (in 32-bit);

D1 is the pulse output port; for 3624MT/2416MT model, only Y0 or Y1 is selectable. Other MT models can only assign Y0/Y1/Y2; the MTQ model can assign Y0/Y1/Y2/Y3/Y4 and etc.

The S2 operating direction output port or the variant can be determined according to S1 and the difference compared with current position. When the output is ON, it means the system is operating in the forward direction, and reverse direction vice versa.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D1		✔													
D2		✔	✔	✔											

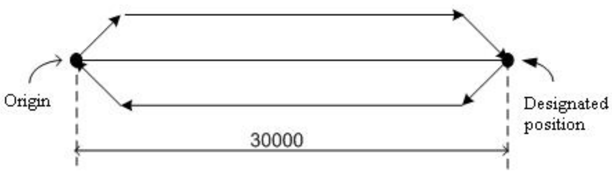
Even if the operand contents are being changed during the instruction execution process, it will not show the effect in the currently running operation. The change will only become effective in next instruction execution.

When the instruction-driven contacts become OFF during the execution process, the machine will start to decelerate and eventually stop. The completion signal of M8029 will be executed at this time and not further action will be carried out.

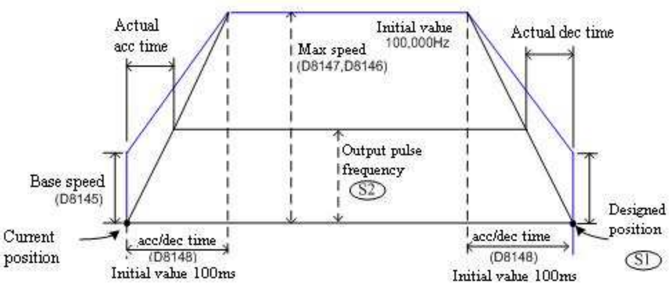
When the instruction-driven contacts become OFF and the pulse output interruption signals, M8147 (Y000), M8148 (Y001) are on, the re-initiating instruction will not be accepted.

Programming Example:

The instruction is a type of control method to control the operating movement of machinery from the assigned origin toward the designated point.



During the pulse output process, the frequency will either accelerate or decelerate according to the preset value.



The actual minimum pulse output frequency is determined according to the following formula:

Minimum pulse output frequency = $\sqrt{\text{Max speed[D8147,D8146]HZ}\div(2\times\text{Acc/dec time[D8148]ms}\div1000)}$

Even if the assigned value is lower than the above calculated result, the frequency to be exported will still be the calculated value. The frequencies in the initial stage of acceleration and in the final section of deceleration must not be lower than the above calculated result. During the instruction execution, the involved system variables are as follows:

［D8145］: Base speed when executing FNC158 (DRVI) and FNC159 (DRVA) instructions. During the operation of stepping motor, the stepping motor's resonance region and automatic start frequency must be considered when setting up the speed. Setting Range: below 1/10 of the highest speed (D8147, D8146). When the setting surpasses the indicated range, the operating speed will automatically decelerate to the 1/10 of the highest speed.

[D8147 (high byte), D8146 (low byte)]: Maximum speed when executing FNC158 (DRVI) and FNC159 (DRVA) instructions. The assigned output pulse frequency must be lower than the maximum speed. Setting range: 10 ~100 , 000 （Hz）

[D8148]: acceleration and deceleration time when executing FNC158 (DRVI) and FNC159 (DRVA) instructions. Acceleration/Deceleration time means the time required in order to reach the maximum speed (D8147, D8146). Therefore, when the output pulse frequency is lower than the maximum speed (D8147, D8146), the actual acceleration/deceleration time will reduce. Setting range: 50 ~ 5，000 (ms)

［M8145］ : Y000 pulse output stopping (immediate stopping)

［M8146］ : Y001 pulse output stopping (immediate stopping)

［M8152］ : Y002 pulse output stopping (immediate stopping)

［M8153］ : Y003 pulse output stopping (immediate stopping)

［M8154］ : Y004 pulse output stopping (immediate stopping)

［M8147］ : Y000 pulse output monitoring (BUSY/READY)

［M8148］ : Y001 pulse output monitoring (BUSY/READY)

［M8149］ : Y002 pulse output monitoring (BUSY/READY)

［M8150］ : Y003 pulse output monitoring (BUSY/READY)

［M8151］ : Y004 pulse output monitoring (BUSY/READY)

Notice:

Positioning instruction (ZRN/PLSV/DRVI/DRVA) can be reused in the program, but do not output to the same port;

If the drive power flow for an instruction turns OFF and ON again, it can only be driven after one operation cycle when status bit (Y000: [M81471], Y001: [M8148] ,Y002: [M8149], Y003:[M8150], Y004: [M8151]) turns OFF.

When positioning instruction is driven again, there should be at least one cycle of OFF time. If the re-drive is implemented in the time less than above condition, there will be calculation error when firstly implementing calculation instruction.

Note:

In the new H2U series PLC, improvements the functions of PLSR, DRVI, and DRVA instructions are introduced. Please refer to Appendix 8.7 in the Programming Manual.

DRVI instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DRVI	Relative Positioning	16	No	DRVI S1 S2 D1 D2	9
DDRVI		32	No		17

Based on the assigned port, frequency, and the assigned pulse output value of the operating direction, the instruction allows machines to perform offset movement according to its present position. Only the PLC with the transistor output can execute the instruction, where:

S1 represents the assigned output pulse value. When commend is in 16-bit, the range is from -32,768 to 32,767; when it is in 32-bit, the range is from -2,147,483,648 to 2,147,483,647. The negative symbol indicates the opposite direction.

(S2) represents the assigned output pulse frequency. When instruction is in 16-bit, the range is 10~32,767Hz; when in 32-bit, the range is 10~100,000Hz;

(D1) is the pulse output port; for 3624MT/2416MT model only Y0 or Y1 can be assigned. Other MT models can only assign Y0/Y1/Y2; the MTQ model can assign Y0/Y1/Y2/Y3/Y4, and etc;

(D2) is the operating direction output port or variant. When the output is in ON state, the system is operating in the forward direction, and vice versa.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D1		✔													
D2		✔	✔	✔											

Output pulse value is treated as the relative position when comparing with the current value of the register described below:

When exporting to [Y000], the current register value is [D8141 (high byte), D8140 (low byte)] (in 32-bit).

When exporting to [Y001], the current register value is [D8143 (high byte), D8142 (low byte)] (in 32-bit).

When exporting to [Y002], the current register value is [D8151 (high byte), D8150 (low byte)] (in 32-bit).

When exporting to [Y003], the current register value is [D8153 (high byte), D8152(low byte)] (in 32-bit).

When exporting to [Y004], the current register value is [D8155 (high byte), D8154 (low byte)] (in 32-bit); when in reverse, the current value of the register reduces.

Even if the operand contents are being changed during the instruction execution process, it will not show the effect in the currently running operation.

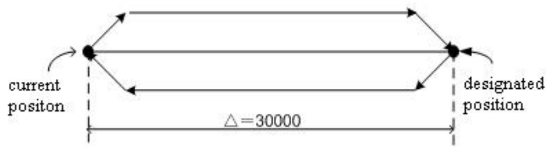
When the instruction-driven contacts become OFF during the execution process, the machine will start to decelerate and eventually stop. The completion signal of M8029 will be executed at this time and not further action will be carried out.

After the instruction-driven contacts become OFF, and the pulse output interruption signals M8147 (Y000), M8148 (Y001), M8149 (Y002), M8150 (Y003), and M8151 (Y004) are in ON state, re-initiation instruction will not be accepted.

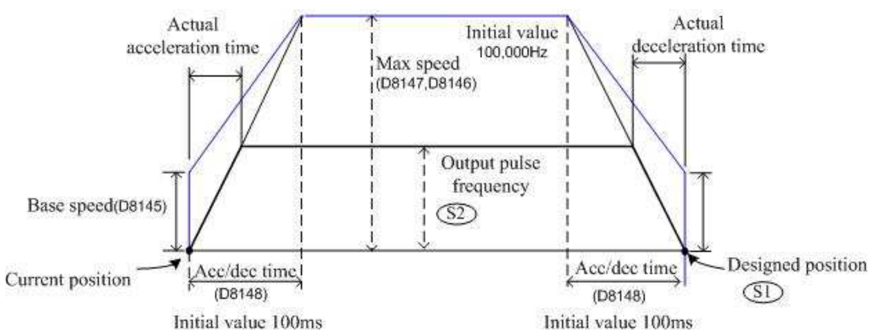
Programming Example:

```
         S1       S2      D1   D2
M11
─┤├──(DRVI  K30000   K4000   Y0   Y3)
```

With 30000 pulses exported from the Y0 port at the frequency of 4 kHz, the external server allows the machine to operate in directions that are determined by Y3.



During the pulse output process, the frequency will either accelerate or decelerate according to the preset value.



The actual minimum pulse output frequency is determined according to the following formula:

Minimum pulse output frequency = $\sqrt{\text{Max speed[D8147,D8146]HZ} \div (2 \times \text{Acc/dec time[D8148]ms} \div 1000))}$

Even if the assigned value is lower than the above calculated result, the frequency to be exported will still be the calculated value.The frequencies in the initial stage of acceleration and in the final section of deceleration must not be lower than the above calculated result.

During the instruction execution, the involved system variables are as follows:

［D8145］: Base speed when executing FNC158 (DRVI) and FNC159 (DRVA) instructions. During the operation of stepping motor, the stepping motor's resonance region and automatic start frequency must be considered when setting up the speed. Setting Range: below 1/10 of the maximum speed (D8147, D8146). When the setting surpasses the indicated range, the operating speed will automatically decelerate to the 1/10 of the highest speed.

[D8147 (high byte), D8146 (low byte)]: Maximum speed when executing FNC158 (DRVI) and FNC159 (DRVA) instructions. The assigned output pulse frequency must be lower than the maximum speed. Setting range: 10 ~100 , 000 （Hz）

[D8148]: acceleration and deceleration time when executing FNC158 (DRVI) and FNC159 (DRVA) instructions. Acceleration/Deceleration time means the time required in order to reach the maximum speed (D8147, D8146). Therefore, when the output pulse frequency is lower than the maximum speed (D8147, D8146), the actual acceleration/deceleration time will reduce. Setting range: 50 ~ 5,000 (ms)

［M8145］ : Y000 pulse output stopping (immediate stopping)

［M8146］ : Y001 pulse output stopping (immediate stopping)

［M8152］ : Y002 pulse output stopping (immediate stopping)

［M8153］ : Y003 pulse output stopping (immediate stopping)

［M8154］ : Y004 pulse output stopping (immediate stopping)

［M8147］ : Y000 pulse output monitoring (BUSY/READY)

［M8148］ : Y001 pulse output monitoring (BUSY/READY)

［M8149］ : Y002 pulse output monitoring (BUSY/READY)

［M8150］ : Y003 pulse output monitoring (BUSY/READY)

［M8151］ : Y004 pulse output monitoring (BUSY/READY)

Notice:
Positioning instruction (ZRN/PLSV/DRVI/DRVA) can be reused in the program, but do not output to the same port;
If the drive power flow for an instruction turns OFF and ON again, it can only be driven after one operation cycle when status bit (Y000: [M81471], Y001: [M8148] ,Y002: [M8149], Y003:[M8150], Y004: [M8151]) turns OFF.
When positioning instruction is driven again, there should be at least one cycle of OFF time. If the re-drive is implemented in the time less than above condition, there will be calculation error when firstly implementing calculation instruction.

Note: In the new H2U series PLC, improvements the functions of PLSR, DRVI, and DRVA instructions are introduced. Please refer to Appendix 8.7 in the Programming Manual.

DSIN instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DSIN	Floating point SIN calculation	32	No	DSIN S D	9
DSINP		32	Yes		9

The instruction is used to calculate the SIN value of the designated angle (RAD, radian). The variables are in the storage format of binary floating points, where:

S is the angle variable that needs to be calculated in order to obtain SIN value. The unit is in RAD, and the value is expressed in binary floating points.Value Range 0<=α<=2π;

D is the storage unit for the SIN calculation results after its conversion. It is in binary floating point format.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
D													✔		

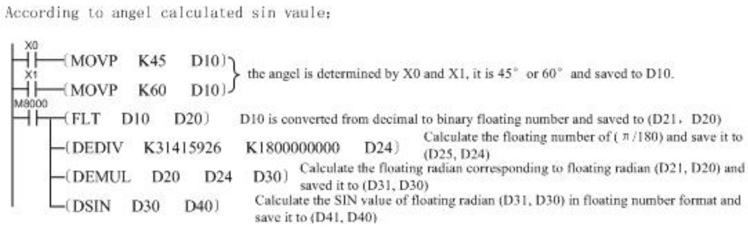
Programming Example:

Example 1 for instruction:

The calculated source data and SIN results are all in binary floating point value format.

RAD(radian)value＝angle×π/180°,for example, the radian corresponding to angle 360°＝360°×π/180°＝2π.

Example 2 for instruction:



DSW instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DSW	Digital Switch	16	No	DWS S D1 D2 n	9

The instruction is used to read the status of matrix-setting switch. One set includes four BCD setting switches. After settings are read, they will be saved in the designated units. Two are the maximum number of switch sets that can be read, where

S is the starting port button of scanning input X port. If n=1, the four succeeding X ports will be used; if n=2,=2, the eight succeeding X ports will be used.

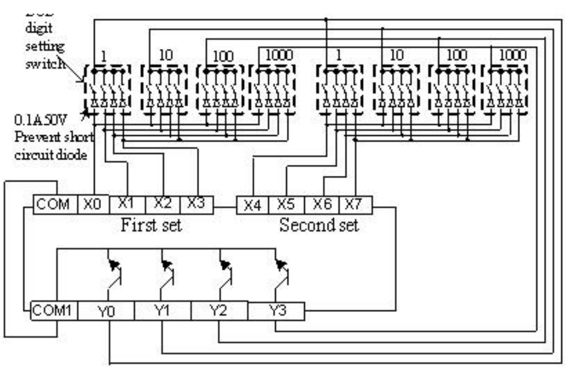
D1 is the starting port button of scanning output Y port, and it uses the four succeeding Y ports.

D2 is the input value storing unit, 0~9999;

n is the number of switch set. Only 1~2 can be selected.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔														
D1		✔													
D2											✔	✔	✔	✔	✔
n	Constant，n＝1~2														

Programming Example:

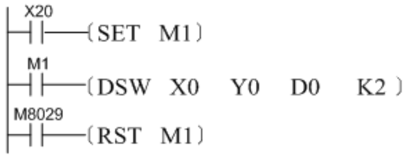


Perform the operation to scan and read the digit switch setting if X 20=ON.
1. The setting values for the first set of digit switches are converted to BIN and saved to D 0;
2. The setting values for the second set of digit switches are converted to BIN and saved to D 1;
3. M8029 will be set for one scanning cycle after one-time reading is completed.

Instruction for use:

Only the PLCs with transistor outputs can detect the digital switch.

- The READ operation of one digital switch requires multiple scanning cycles to complete. If the READ operation is activated using buttons, it is recommended to use the following programming statements to ensure the readable cycle's integrity.



DTAN instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DTAN	Floating point TAN calculation	32	No	DTAN S D	9
DTANP		32	Yes		9

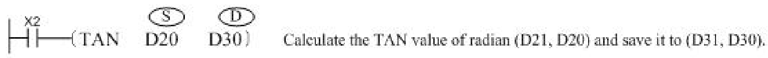
The instruction is used to calculate the TAN (tangent) value of the designated angle (RAD, radian). The variables use the binary floating point storage format.

S is the angle variable that needs to be calculated to obtain the TAN value. The unit is in RAD, and the value is expressed in binary floating points. Value Range $0<=\alpha<2\pi$;

D is the storage unit for the TAN calculation results after its conversion. It is in binary floating point format.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
D													✔		

Programming Example:



The calculated source data and SIN results are all in binary floating point value format.

RAD(radian)value＝angle×π/180°,for example, the radian corresponding to angle 360°=360°×π/180°=2π.

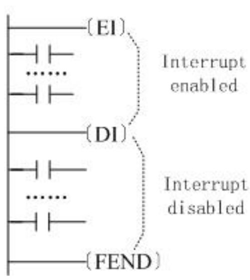
In regards to the programming statements used to calculate the TAN value, please refer to the example in the SIN instruction section.

EI、DI instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
EI	Permissible Interruption	16	No	No operand, independent instruction that does not require initiating contacts	1
DI	Prohibitive Interruption	16	No		1

When PLC program is in operation, interruption is prohibited; after the execution of EI instruction, interruption function has become permissible; when DI instruction is executed during the permissible interruption status, the system enters the status of prohibitive interruption. In the programming, if there is no inter-zoning interruptive prohibition insertions, DI instruction cannot be executed.



Interrupt types and setting:

1） External signal input interrupts: they can be defined to trigger interrupts by rising or falling edges. For an X signal that doesn't need an immediate response, pulse capture function can also be used;

2） Timer interrupts: they occur every fixed period of 1ms~99ms.

3） High speed interrupts: they are used with DHSCS comparison setting instruction. Interrupt occurs when the present value of a high speed counter reaches the setting value.

External Signal Input Interruption Indication and Setup:

Input No.	Pointer No.		Interrupt disabled instruction
	Rising edge interrupt	Falling edge interrupt	
X000	I001	I000	M8050
X001	I101	I100	M8051
X002	I201	I200	M8052
X003	I301	I300	M8053
X004	I401	I400	M8054
X005	I501	I500	M8055

Timing Interruption Indicator and Setup:

Input No.	Interrupt period (ms)	Interrupt disable instruction
I6□□	Input 1~99 to □□ in the instructions, for example, I605, which executes one timing interrupt every 5 ms	M8056
I7□□		M8057
I8□□		M8058

High-speed interruption indicator and setup:

Input No.	Interrupt disable instruction
I010	M8059
I020	
I030	
I040	
I050	
I060	

Pulse Output Completion and Interruption Indicator and Setup: (the function requires the activation of M8090~M8094 in order to generate interruption after pulse output has been completed)

Port No.	Use special bit	Related user interrupts
Y000	M8090	I502
Y001	M8091	I503
Y002	M8092	I504
Y003	M8093	I505
Y004	M8094	I506

Interrupting sub-program uses different numbers to select different ports and interruption trigger edge;

External input interrupt can only be applied on same X, and it cannot be applied to both ascension and descension interrupting numbers at the same time. Only one trigger edge can be applied to one X input port. The trigger edge can be configured through indicator numbers.

External input interrupt: if M8050-M8055 is in the status of "ON" during the program execution process, the interruption function of the corresponding X port is prohibited.

Timing Interruption: if M8056-M8058 is in the status of "ON" during the program execution process, the interruption function of the corresponding X port is prohibited.

High-speed counter interruption: if M8059 is in the status of "ON" during the program execution process, the interrupting function of all the high-speed counters is prohibited.

Interruption instruction's programming requirements and execution features:

- Interruptions can be applied in between the D1 and E1 instructions (between the zones of prohibitive interruptions). The instruction can be saved in memory and later on executed after the EI instruction.
- Indicator number cannot be reused.
- When multiple interruptions are occurring in sequence, the prioritization is based on the sequence. When interruptions are happening all at

the same time, the priority will base on it level of classification. The priorities from high to the low end are: high-speed counter, external, timing, pulse output completion.

- During the interruption execution process of regular programs, other interruptions are prohibited. However, if EI and DI instruction programs are being edited under the interruptive sub-programs, a maximum of two interruptions can be programmed.

- During the interruption process, both input and output relays can be controlled. By executing the input/output refresh instruction (REFF), the most current input status can be read, and the calculation results can be exported immediately to realize the task of high-speed control.For input relay numbers that are to be used by interruption indicator, please do not use numbers that are used in application instructions such as [high-speed counter] and [pulse density], which choose from the same input range.

- For the timer used in sub-programs and routine interruption programs, please use the T192-T199 timer specifically for the routine program. Should other regular timers be used, not only it cannot carry out the timing function, extra caution must be paid when using the 1ms cumulative timer.

- If the input interruption indicators, I and 0 ports, are designated, the input filter feature of the input relay will be automatically shut off. Therefore, it is unnecessary to use the REFE instruction and the special data register D8020 (input filter adjustment). Besides, the input filter of the input relay that is not being used by the input interruption indicator can maintain for 10ms (initial value).

In order to satisfy the operation of the high-speed counter, 30 additional high-speed counting interruptions are added. This allows any designated high-speed interruption to produce 30 interruption responses. The function is called "Multiple User-designated Interruption Feature" of high-speed counter. The operating configuration follows the following patterns:

Flag bit	description
M8084	Set to ON to enable high speed counter multi-user interrupts
D8084	High speed counter no. C235~C255
D8085	Related user interrupt numbers, 24 max from I507 to I530
D8086	Correspond to the serial numbers of several compare point data and can be used as D component only, such as 200, which represents a double word starting from D200.

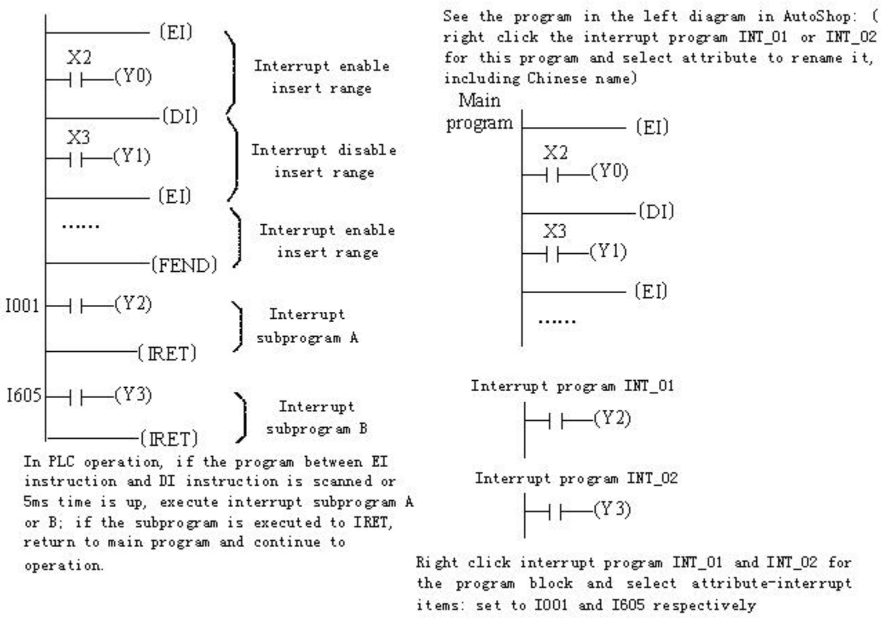
Example of the Comparison Point Data Storage:

D8084=235;D8086=200;D8085=5;M8084=ON;

The data in C235	Recording unit	Save unit value	Related user interrupts	Value in D8131
100	D200、D201	=100	I507	0
200	D202、D203	=200	I508	1
300	D204、D205	=300	I509	2
400	D206、D207	=400	I510	3
500	D208、D209	=500	I511	4→0(M8133=ON)

Every interruption can be produced by the values in the high-speed counter and the recorded units.

Programming Example



EMOV instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEMOV	Binary floating-point data transmission	32	No	EMOV S D	9
DEMOV		32	Yes		9

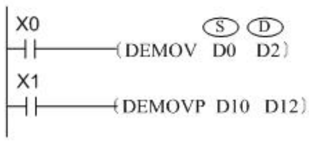
Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

The instruction performs index calculation using binary floating-point data on the basis of e (2.71828). Where:

S the binary floating-point variables that is to be used to calculate binary floating-point index;

D the storage unit for index calculation results.

Programming Illustration



If the binary floating value in (D1, D2)= 12.345, when X0=ON, the binary floating value in (D3, D2) will become 12.345. When M0 is changes from ON to OFF, (D3, D2) remains 12.345 unless user program changes (D3, D2) again. Or (D3, D2) will be changed only when PLC is switched from STOP to Run or powers up. The value remains the same whether power failure hold register powers up or PLC is switched from STOP to RUN.

ENCO instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ENCO	Programming (obtain the data's ON position and convert it to BIN data)	16	No	ENCO S D n	7
ENCOP		16	Yes		7

Calculate n's position value in S as the bit indicator. Use the corresponding position of D as 1, and others are all 0.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔	✔	✔	✔							✔	✔	✔	✔	✔
D											✔	✔	✔	✔	✔
n	Constant, n=1~8. When n=0, the instruction will not execute; other values will be debugged.														

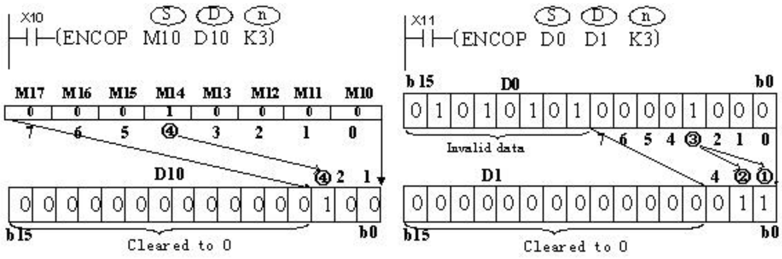
When there are multiple positions in the source address that has the value of 1, only the position at the high end side with the first 1 will be calculated; Error message will appear when all positions of S are 0;

When driver input is OFF, the instruction will not be executed, and the output number will not change.

When n=8, if the programming instruction S is a digital component, it has 256 dots.

The instruction usually uses pulse operation type instruction.

Programming Illustration:



EXP instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DEXP	The instruction performs index calculation using binary floating-point data on the basis of e (2.71828).	32	No	EXP S D	9
DEXPP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

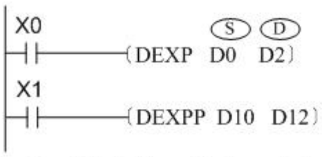
S is the transmission source of Binary floating-point data

D is the storage unit for saving binary floating point data

Requires contact driver, and there are 2 operating variables. Copy the binary floating point data from S to D.

Note: When the calculation result is not within the range of $[2^{-126} \leqslant \text{Operation result} < 2^{128}]$ calculation error message will appear. Error code is K6706,it saved in D8067, set ON at M8067 which is sign bit of error.

Programming Illustration



When X0=ON, the binary floating value in (D1, D0) is implemented with index calculation and then saved to (D3, D2).e (D1、D0) (D3、D2)

Since loge2128=88.7, D8067= K6706 and M8067 is set to ON when (D1, D0) is higher than 88.7.

FLT instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
FLT	Converting BIN from integrals to binary floating points.	16	No	FLT S D	5
INTP		16	Yes		5
DFLT		32	No		9
DFLTP		32	Yes		9

The instruction coverts the integral S to floating digits, and saves the result in D and D+1 units.

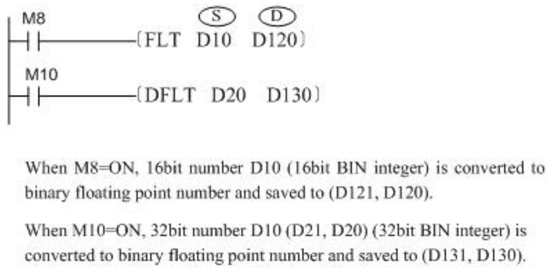
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
D													✔		

Constants K and H will be converted automatically in every floating point calculation instruction. Therefore, FLT instruction cannot be used here.

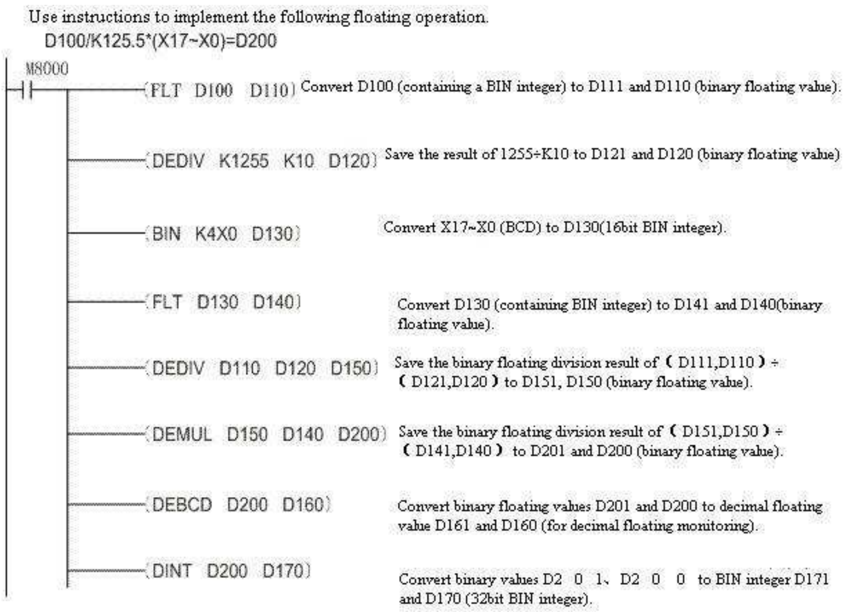
The instruction's inverse transformation instruction is INT (converts binary floating point values to BIN integrals).

Programming Illustration

Example 1 for instruction:



Example 2 for instruction:



FMOV instruction

Instruction description:

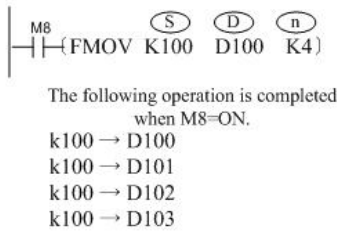
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
FMOV	Multiple Point	16	No	FMOV S D n	7
FMOVP		16	Yes		7
DFMOV		32	No		13
DFMOVP		32	Yes		13

It requires contact driver, and it has three operating variables. S data will be copied and saved in the n units, which the start address is designated by D.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔		

D								✔	✔	✔	✔	✔	✔		
n	Constant,n＝1~512														

Programming Illustration



FOR,NEXT instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
FOR	Begin the Cycle Range	16	No	FOR S1	3
NEXT	End the Cycle Range	-	-	NEXT (no operand)	1

FOR is used to begin a cycle and indicate the number of times the cycle will be executed. It must be used together with the NEXT instruction. Where: S1 is the cycle frequency controlling variable.

NEXT Instruction is used to indicate the end portion of the cycle. FOR instruction designates the number of cycles of FOR~NEXT to be repeated. After the cycles have been completed, it will exit from the FOR~NEXT cycle and continue the operation.

Between each cycle of FOR~NEXT instructions, another FOR~NEXT cycle may be inserted. However, the condition is, calculating from the outermost FOR~NEXT cycle, only 4 FOR~NEXT cycles can be inserted. During the operation, PLC will analyze and execute FOR~NEXT based on each corresponding cycle. Please note that when there are excessive numbers of cycles, the PLC scanning cycle will be prolonged. This may cause the timeout monitor timer to activate and lead to potential errors.WDT instruction may be implemented between the FOR~NEXT instruction to prevent errors.

Error messages will appear under the following situations:

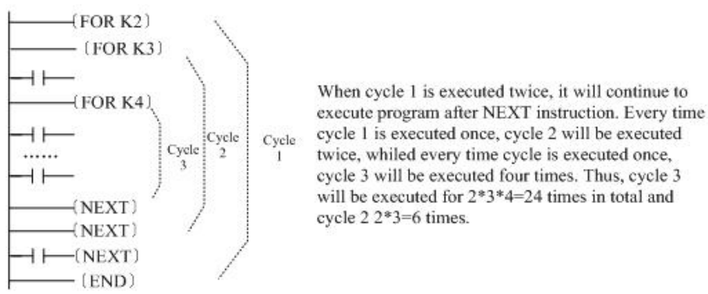
- NEXT instruction is programmed prior to FOR instruction;
- FOR instruction is programmed but not NEXT instruction;

Disagreement between the numbers of FOR and NEXT instructions.

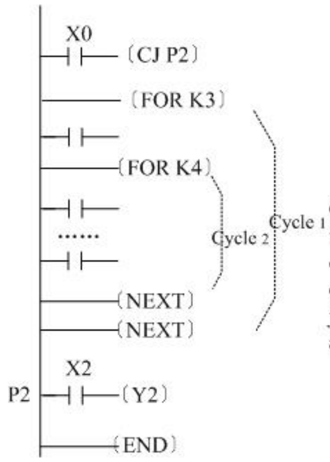
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

Programming Illustration

Example 1 for instruction:

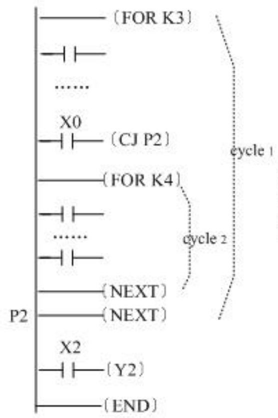


Example 2 for instruction:



CJ instruction can be used to skip FOR~NEXT instruction. In the example, if X0=OFF, execute cycle 1 and 2; if X0=OFF, executed cycle 1 and 2; when X0= ON, CJ instruction jumps to P2 and the program between cycle 1 and cycle 2 will not be executed.

Example 3 for instruction:



CJ instruction can also be used to skip inner nesting FOR~NEXT instruction. In the example, if X0=OFF, execute cycle 1 and 2; if X0=ON, CJ instruction jumps to P2 and cycle 2 nested inside cycle 1 will be skipped.

FROM instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
FROM	BFM Read out	16	No	FROM m1 m2 D n	9
FROMP		16	Yes		9
DFROM		32	No		17
DFROMP		32	Yes		17

The instruction is used to read the data retrieval operation of the BFM register in the special extended module.

m1 is the address serial number of the special extended module, whose value ranges 0~7. 0 is the closest to the main module and the number goes on. Maximum of 8 special modules are allowed.

m2 is the register address code of BFM inside the special module. It has values ranging from 0~32767;

D is the storage address after reading the parameters in the main module. When the number of register read is more than one, it occupies the following units.

n The number of parameters read during the operation (counted by Word). It has values ranging from 1~32767. The values will be read in sequence according to the register addresses.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D								✔	✔	✔	✔	✔	✔	✔	✔
m1=0~7; m2=0~32767; n=1~32767; When designating D component, instructions in 16-bit can use K1~K4; instructions in 32-bit can use K1~K8; m1, m2, and n do not support character devices and D registers.															

Programming Illustration

X0 —(FROM K1 K20 D200 K1) [m1 = K1, m2 = K20, D = D200, n = K1]

When X0 is ON, retrieve the content of the twentieth address (in 16-bit) in #1 special module to the D200 register. One retrieval at a time (n=1). When X0 is OFF, no operation will be executed.

When using instructions in 32-bit, addresses designated by D are the low 16-bit addresses; addresses designated by D+1 are the high 16-bit addresses.

GBIN instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
GBIN	Gray Code's Inverse Transformation	16	No	GBIN S D	5
GBINP		16	Yes		5
DGBIN		32	No		9
DGBINP		32	Yes		9

The instruction converts the GRY Gray Code to binary values. Where:

S is the GRY data source or data variable unit is to be converted; when instruction is in 16-bit the range is 0~32,767; 32-bit 0~3,147,483,647. When the value exceeds the indicated ranges, M8067 and M8067 will be set in ON, and the instruction will not execuate.

D is the storage unit after the value has been converted to BIN.

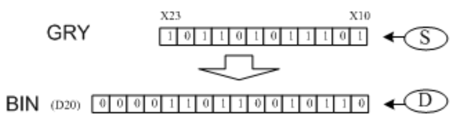
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

GRY→BIN mathematical calculation: from the second digit from the left, calculate every digit with the decoded digit left to it, and use the value of the digit as the decoded value (the far left digit remains unchanged).

Programming Illustration

X0 —(GBIN K3X10 D20) [S = K3X10, D = D20]

Example:



GRY instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
GRY	An instruction used to send original data sample	16	No	GRY S D	5
GRYP		16	Yes		5
DGRY		32	No		9
DGRYP		32	Yes		9

This instruction is used to convert BIN values to GRY codes. Where:

S is the BIN data source or data variable unit to be converted. M8067 and M8068 will be On and the instruction will not be executed if?exceeds the range, 0~32,767 for 16bit instructions and 0~2,147,483,647 for 32bit instructions.

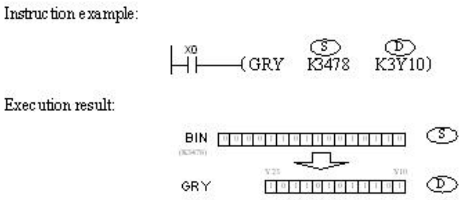
D is the unit where converted GRY code is stored.

	Bit component	Word component

Operand	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

BIN→GRY mathematical algorithm: starting with the rightmost bit, sequently perform XOR operation for that bit to its left bit, and the resulted value will be the GRY value of that bit. While the leftmost bit stays unchanged (i.e. the left is 0);

Programming Illustration



HEX instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
HEX	conversion of	16	No	HEX S D n	7
HEXP	HEX	16	Yes		7

This instruction is used to convert the value of the starting variable of S into an ASCII code and store it in an address starting with D. The number of chars and storage mode can be set by the user. Where:

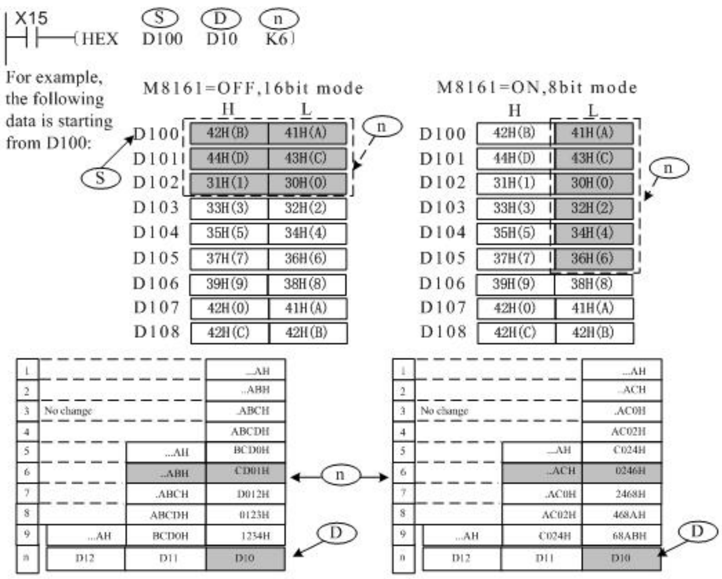
S is the variable address or constant to be converted. If it is a register variable, the conversion interval will has a width of a 32bit variable (i.e. 4 ASCII chars);

D will be converted into the starting address for storing the ASCII code, for which the variable space taken is depending on n.

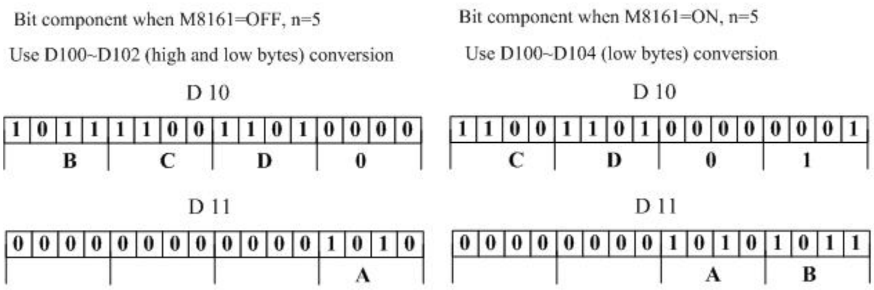
n is the number of bits of the converted ASCII chars.

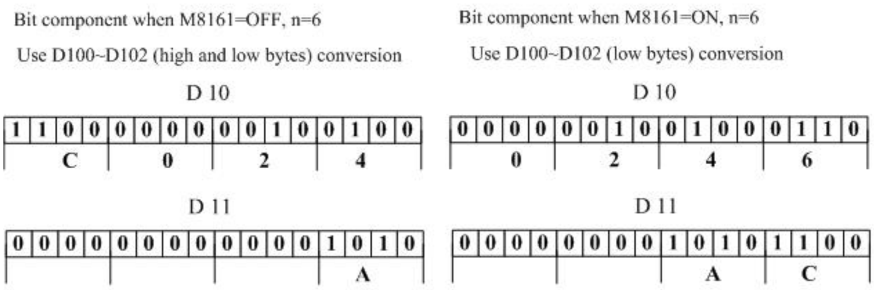
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔
n	Constant, n=1~256														

Programming Illustration



Where the mode of variable width is determined by the M8161 sign: M8161=OFF indicates 16bit mode, which means that both the upper byte and the lower byte are participating the operation; M8161=ON indicates 8bit mode, which means that only the lower byte is participating the operation and the upper byte will be abandoned, as a result the length of the actually used variable area S will increase.





Note:

It should be aware that the M8161 mode sign is shared by instructions of RS/HEX/ASCI/CCD, etc.

The source data of S data area must be of ASCII chars, otherwise errors will occur during the conversion.

If the output data if of BCD format, a BCD-BIN conversion is required after HEX convers to get the correct value.

HKY instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
HKY	16-key input	16	No	HKY S1 D1 D2 D3	9
DHKY		32	Yes		17

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1	✔												✔		
D1		✔													
D2										✔	✔	✔	✔	✔	✔
D3		✔	✔	✔											

This instruction is used to read a 4×4 matrix of 16 keys, which are the decimal 0~9 keys and the functional keys of A~F sequentially. When the keys are pressed (ON), decimal numbers of 4 bits between 0~9999 or functional keys between A~F can be entered, depending on the sequence of the keypress actions. If 32bit instructions are used, decimal numbers of 8 bits between 0~99,999,999 or functional keys between A~F can be entered. Where:

S is the number of the starting port of the scanning input port X of the keys, 4 X ports starting with which will be used;

D1 is the starting port button of scanning output Y port, and it uses the four succeeding Y ports.

D2 is the storage unit for the entered values from the keys, with a range of 0~9999. If 32bit instructions are used, decimal numbers of 8 bits between 0~99,999,999 can be entered.

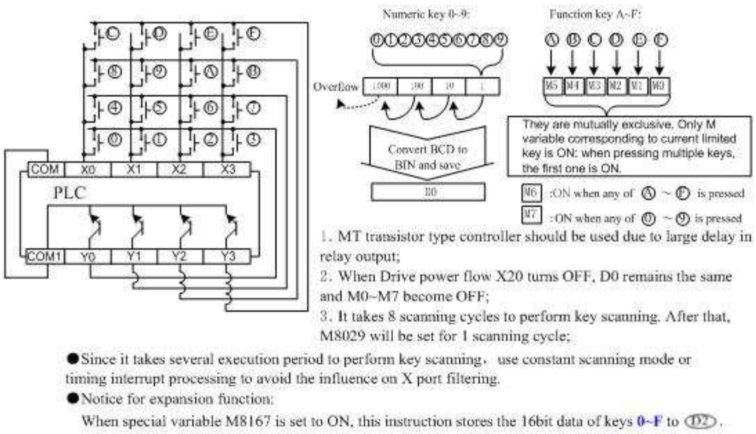
D3 is the address of the starting unit of the entering status of the keys, which occupies a variable unit of 8 continuous bits.

This instruction can only be used for transistor-output type PLC.

Programming Illustration

X20 —(HKY S:X0 D1:Y0 D2:D0 D3:M0)

Corresponding wiring diagram and parameter response instruction as follows:



HOUR instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
HOUR	Timer	16	No	HOUR S D1 D2	7
DHOUR		32	No		13

S This instruction is used to record the accumulative time during which the driving conditions are met. When the set time value is reached, the instruction output is activated. Where:

D1 is the starting unit of the accumulative time;

D2 indicates that the time has reached the warning output variable unit. When the set value is reached, the status specified for this unit is effective.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D1													✔		
D2		✔	✔	✔											

The setting range of D1is K0~K32,767 (in hours) for 16bitD1+1 is the current time value that is less than an hour, with a setting range of K0~K3,599 (in seconds). At this time D1occupies 2 units.

The setting range D1+1 and D1 is K0~K2,147,483,647 (in hours) for 32bit. D1+2 is the current time value that is less than an hour, with a setting range of K0~K3,599 (in seconds). At this time D1 occupies 2 units.

Negative values are not applicable to instructionD1 timing. If D1 is set as the register area for non power failure holding, the value of D1 will be zeroed when the PLC status is changed from STOP to RUN or at a power failure. If the data of the current value need to be kept even under the situation of PLC power failure, please setD1 as the register area for power failure holding.

Programming Illustration

M200 —(HOUR S:K2000 D1:D300 D2:Y10)

When M200=ON, the time this status is holding will be accumulated, the hour value is recorded in D300 and the second value less than 1 hour is recorded in D301. Y10 output status is switched ON when the accumulative time of D300 reaches 2000 hours. If the timing conditions are met, the accumulative timing continues and the readout value will continue to increase when the specified S value is reached. The timing will stop when the current time value of D300 reaches the maximum value of 32,767 hours and D301 reaches 3,599 seconds. You have to zero the current time values of D300 and 310 to restart a new timing.

INC instruction

Instruction Description

This guide specifies the plus 1 (+1) operation of the soft component data.

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
INC	BIN add 1	16	No	INC D	3
INCP		16	Yes		3
DINC		32	No		5
DINCP		32	Yes		5

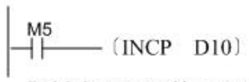
The value of D increases by 1 after each process.

For 16bit operations, 32,767 plus 1 gets -32,768.

For 32bit operations, 2,147,483,647 plus 1 gets -2,147,483,648.

This step will not refresh the "0" sign or the carry and borrow sign.

Programming Illustration



D10 is incremented by 1 every time M5 is set.

INCD instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
INCD	Increment method of cam control	16	No	INCD S1 S2 D n	9

This instruction is used to perform multi-segment comparisons for cam control. The table, timer, etc. used for the comparisons are all set incrementally. The instruction is implemented in the scanning main program, and the comparison result is effected by scan time delay. Where:

S1 is the comparison table.

S2 is the timer. The neighboring S2 +1 unit is used to reset the time on the counter after the caluculation and comparison process. (32bit counters are applicable to 32bit instructions)

D is the comparison results record, which is a bit variable unit occupying n continuous addresses.

n is the number of multi-segment comparison sets.

all point to 32bit variables when using 32bit instructions. S1 S2 D is also calculated based on the length of 32bit variables. When the set comparison of n is done, the "instruction done" flag "M8029" will automatically switch on one scanning cycle.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1							✔	✔	✔	✔	✔	✔	✔		
S2												✔			
D		✔	✔	✔											
n	Constant, n＝1~64;														

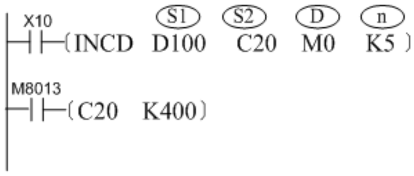
For 16bit - S1 operation numbers KnX, KnY, KnM and KnS, "K4" must be specified.

For 32bit -"K8" must be specified and the number of components X, Y, M and S must be in multiples of 8.

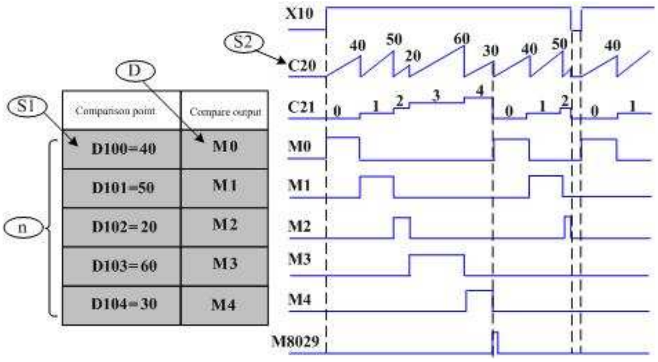
S1 operation numbers are limited to C0~C199 for 16bit instructions.

(S1) operation numbers are limited to C200~C254 for 32bit instructions.

Programming Illustration



If the relevant variables have been set as follows, when X10=ON, the implementation result is shown in the following figure.



Instruction for use:

- **All the variables of the relevant tables should be assigned using the MOV instruction before beginning the INCD process.**
- **The comparison output is also affected by the delay of the user program scan. Therefore, the HSZ high speed comparison instruction can be used for applications that are needed quickly.**
- **The INCD instruction can only be used once in the program.**

INT instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
INT	Conversion of binary floats to BIN integers	16	No	INT (S) (D)	5
INTP		16	Yes		5
DINT		32	No		9
DINTP		32	Yes		9

This instruction rounds a binary float. The fraction part will be abandoned and the resulted binary value will be stored in (D). Where:

(S) is the binary float variable to be round converted.

(D) is the storage unit of the resulted BIN integer after the conversion.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(S)													✔		
(D)													✔		

If the calculation result is 0, the 0 flag bit (M8020) will be reset.

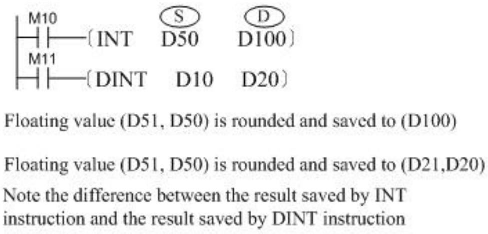
If any fractional part is abandoned during the operation, the borrow sign (M8021) will be set.

If the operation result exceeds the following ranges (overflow), the carry sign (M8022) will be set.

For 16bit instructions:-32,768~32,767

For 32bit instructions:-2,147,483,648~2,147,483,647

Programming Illustration



IST instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
IST	Status initialization	16	No	IST S D1 D2	7

This instruction can be used to initialize the control status of a typical multi-action looping execution mechanism and to specify parameters for the operation mode such as the input signal, action status, etc. Where:

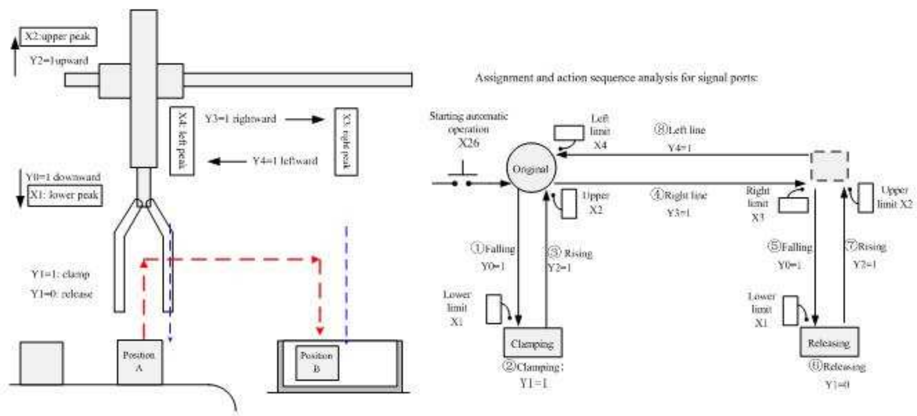
S is the component address of the starting bit variable of the input of the specified operation mode. It occupies 8 continuous address units from S~S+7.The special function definition for each varaible is described below:

D1 is the minimum serial number using the S status in the specified automatic operation mode.

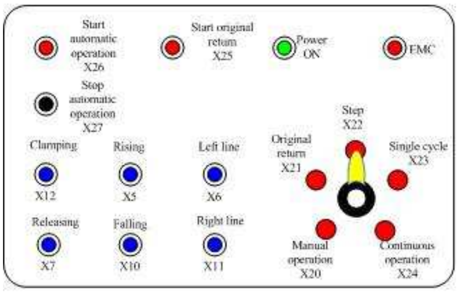
D2 is the maximum serial number using the S status in the specified automatic operation mode.D1 to D2are the status serial numbers of the looping action of the control system, which determine the status numbers.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔	✔	✔												
D1				✔											
D2				✔											
Notice: 1) The instruction is allowed to be used only once in the user program. 2）For D1and D2,only S variables S20 to S899 can be used, and D1must be <D2 3）The special M variable of the system will also be used when using this instruction.															

For example, in the illustrated system below, the execution mechanism acts sequentially in such a way: the grabbing device drops to the position of work piece A from the base point to grab the work piece, and then it lifts the work piece to the specified height and translates to the desired position and drops. After arriving at the required position, it releases the work piece and back tracks to start the next looping action. It is possible to use the IST instruction to specify the control signal input, the control of the status transferring, etc. of the operational mechanism to achieve automatic control. In addition, it supports manual commissioning of single-step actions and base point reset, etc.



Instruction keys and status changing switches are required to control the operational mechanism using manual commissioning, single actions, and looping actions, etc. The following is a schematic diagram of the operation panel, including the key ports and their function assignments:



For applications like the above diagram, each complete cycle can be divided into 8 steps (i.e. 8 statuses). The following instruction clauses can be used to initialize the status of the control system:

```
M8000            S      D1     D2
--| |-----(IST  X20    S20    S27)
```

S specifies X20 as the starting input of the operation mode. Therefore, the input ports X21 to X27 of the subsequent addresses will also be used.The functional action features will be defined respectively as: (it is similar for variables X, M, or Y)

X20 :This is the manual operation mode to switch on/off the various control output signals using a single button.

X21 : This is the base point reset mode to reset the device to the base point by pressing the base point reset button.

X22 : This is the single-step operation mode to step forward a process each time the starting button is pressed.

X23 : The is the one-cycle looping mode. When the start button is pressed, it will run the one-cycle looping automatically and stop at the base point. The operation can be stopped by pressing the stop button. Then, if the start button is pressed, the operation will continue and stop at the base point automatically.

X24 : This is the continuous operation mode to run continuously by pressing the start button. When the stop button is pressed, it will move to the base point and stop.

X25 : To start the base point rest command signal.

X26 : To start the automatic command signal.

X27 : To stop the automatic command signal.

Note: In these port signals, the operation mode is determined by X20 to X24, for which the statuses can't be ON at the same time. Therefore, it is suggested to use rotary switches for the selection and switching of the signals.

D1 and D2 are used to specify the minimum and maximum serial number S20 to S27 of the service statuses (8 for total) in the automatic operation mode. The following special variables for the definition and use requirements of the IST instruction should be noted:

When driving the IST instruction, the control of the following components will be automatically switched and can be referenced by user programs. In order to make the status switching and control of the IST instruction cooperate, the operation of certain special variables is required in the user programs. See the description in the table below:

Default variables in IST instruction		Variables driven in user program	
M8040	1=disable transfer of all states	M8043	1=original return completed. In the original return mode, after a machine returns to original, the special M variable will be set by the user program.
M8041	1= transfer start	M8044	1=original condition. Detect the original condition of a machine and drive the special assistant relay. It is set in all modes.
M8042	1= start pulse	M8045	1=all output reset disabled. If a machine is switched among manual, return and automatic modes when it is not at original, all output and action states will be reset. But only action status can be reset if M8045 has been driven.
S0	Manual operation initial state	M8047	1=STL monitoring valid. After M8047 has been driven, the state number of action (S0 ~ S899) will be saved in the special assistant relay D8040 ~ D8047 in ascendent order, thus monitoring action state numbers of 8 points. In addition, if any of these states is enabled, the special assistant relay M8046 will act.
S1	Original return initial state		
S2	Automatic operation initial state		

- Under the "automatic operation" mode, free conversion is possible between: single step<-->one-cycle looping<-->continuous operation.

- When performing conversion between "manual operation"<-->"base point reset"<-->"automatic operation" while the machine is running, the switched mode is effective after all the outputs are reset. (Reset is not applicable for M8045 drive.)

- S10 to S19 can be used for the base point reset when using the IST instruction. Therefore, don't use these statuses as common statuses. In addition; S0 to S9 are used for the initial status process, S0 to S2, as mentioned in the above manual operations, are used for the base point reset and automatic operation, and S3 to S9 can be used freely.

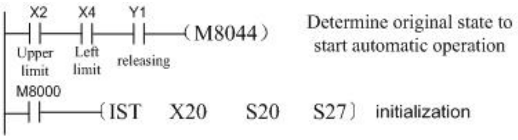
- When programming, the IST instruction must be programmed with a higher priority than the various STL circuit, such as status S0 to S2, etc.

- Rotary switches must be used to avoid the situation that X20 to X24 are ON at the same time.

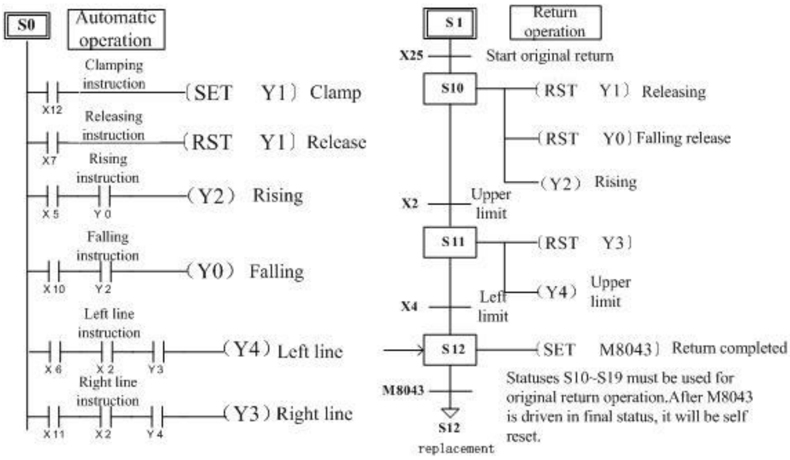
- When switching between each (X20), base point reset (X21), auto (X22, X23, X24) before the base point completion signal (M8043) is activated, all the outputs are switched OFF. And the automatic operation can't drive again until the base point reset is finished.

After initialization of the control instruction using the IST instruction, the action of each status of the execution mechanism and the conditions for status transferring need to be programmed, as detailed below:

1.System initialization: defines the conditions for base point reset and defines the input ports of the operation mode signals used in the IST instruction and the status variables of the looping actions. The program clauses used are illustrated in the following diagram.

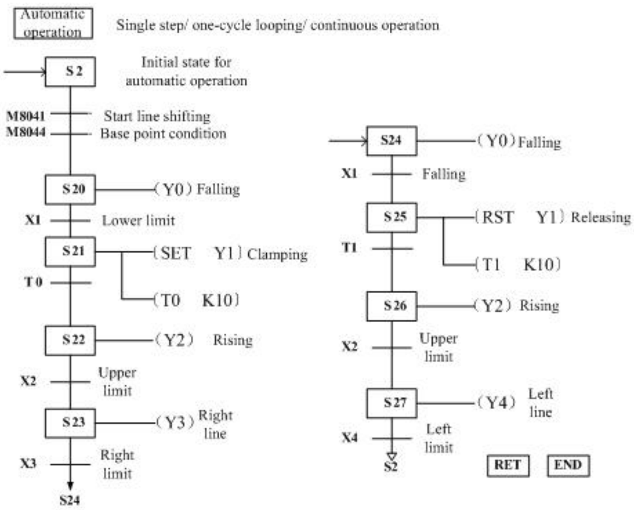


2. Manual operation: driven to execute by the command signals defined on the operation plate. See the program clauses of status S0 in the following diagram. This part of the program can be skipped if there is no manual mode:



3. Base point reset: design reset program based on the command signal at the starting of the reset and the sequence of the reset actions, as shown in the upper right:

4. Automatic operation: write program based on the required action conditions and sequence and the control signal output, as shown in the diagram below:



Up to this point, the control system is allowed to complete the looping action according to the above mentioned action requirements. The above programming description uses step instructions for the convenience of reading, while the user is free to program using the equivalent ladder diagrams.

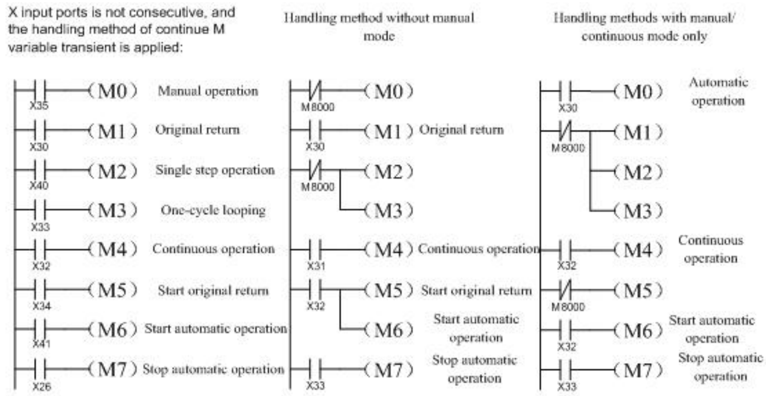
When different status numbers occur to the "automatic operation" mode in a control system, the above example can be referenced to program in modifying the setting items of D1 and D2 corresponding works need to be done in the "automatic operation" mode.

•

Handling methods for non-continuous X input:

If an X input port with non-continuous addresses needs to be used as the provided input of the operation mode, the M variable can be used for a "transitional" transmission. That is, the non-continuous X input status will be copied to an M variable with continuous addresses one by one using the simple OUT instruction rather than the instructions below:

Specific to the continuous M0 to M7 variable area in the IST, the programming instructions can be used to shield the non-existent control mode. For example, the corresponding relationship between X as the mode input end and the M variable in the following diagram. For un-required modes, you simply input the M variable and fix it to zero:



LOGE instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DLOGE	To perform the natural logarithm operation of a binary float to the base e (2.71828)	32	No	LOGE S D	9
DLOGEP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

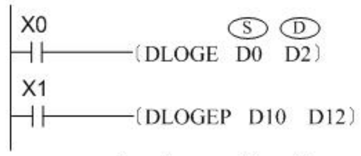
This instruction performs the natural logarithm operation of a binary float to the base e (2.71828). Where:

S is the binary float variable to be calculated for the natural logarithm.

D is the storage unit of the resulted natural logarithm after the operation.

Only a positive value is applicable to the value in S, and a 0 or negative value will cause an error during the calculation. The error code is K6706, which will be saved in D8067, and the error flag bit M8067 is set to ON.

Programming Illustration



When X0=ON, perform the natural logarithm operation of a binary floating point value in (D1, D0) to the base e and save it to (D3, D2).

$$\log_e(D1、D0) \Longrightarrow (D3、D2)$$

The following is the formulas to convert natural logarithm to common logarithm (use 0.4342945 to split the value of common logarithm):

$$10^{x} = e^{\frac{x}{0.4342945}}$$

LOG instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DLOG	To perform the common logarithm operation of a binary float to the base 10	32	No	LOG (S) (D)	9
DLOGP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
(S)					✔	✔	✔							✔		
(D)														✔		

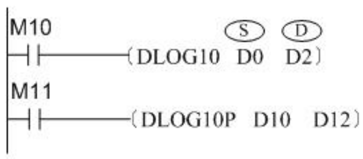
This instruction performs the common logarithm operation of a binary float to the base 10.

(S) is the binary float variable to be calculated for the common logarithm.

(D) is the storage unit of the resulted common logarithm after the operation.

Only a positive value is applicable to the value in (S),and a 0 or negative value will cause an error during the calculation. The error code is K6706, which will be saved in D8067, and the error flag bit M8067 is set to ON.

Programming Illustration



When M10=ON, perform the common logarithm operation of a binary floating point value in (D1, D0) to the base 10 and save it to (D3, D2).

$$\log_{10}(D1、D0) \Longrightarrow (D3、D2)$$

MEAN instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
MEAN	The instruction	16	No		7

MEANP	to calculate the mean value of a data set	16	Yes	MEAN S D n	7
DMEAN		32	No		13
DMEANP		32	Yes		13

This instruction calculates the mean value of the n variables starting with S (sums them up and then divides by n) and stores it into D.

Any remainder occur during the calculation will be abandoned.

An error will occur if the value of n is not in the range of 0~64.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔		
D								✔	✔	✔	✔	✔	✔	✔	✔
n	Constant, n=1 to 64, otherwise an error will occur.														

Programming example:

(D10+D11+D12+D13)/4=D20
If D10=K5, D11K5, D12=K15, D13=K52, then D20=K19, the remainder will be rounded off.

MODBUS instruction

Instruction description:

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
MODBUS	MODBUS transmission	16	No	MODBUS S1 S2 n D	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔							✔		
S2					✔	✔							✔		
n					✔	✔							✔		
D													✔		

S1 is the number and address.

S2 is the address of the register.

n is the number of the registers (16bit number).

D is the data buffer cache.

MOV instruction

Instruction Description

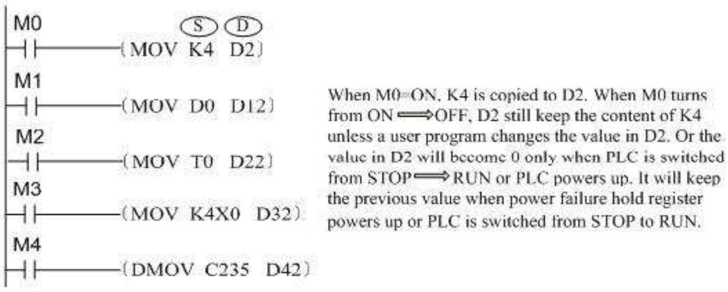
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
MOV	An instruction used to send original data sample	16	No	MOV S D	7
MOVP		16	Yes		7
DMOV		32	No		13
DMOVP		32	Yes		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

A contact drive is required. There are 2 operational variables to copy the value of S into D.

For 32bit instructions (DMOV), both S and D will use the variable unit of the neighboring upper address for the calculation. For example, clause DMOV D1 D5 will result D1→D5 and D2→D6.

Programming example:



MTR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
MTR	Matrix input	16	No	MTR S D1 D2 n	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔														
D1		✔													
D2		✔	✔	✔											
n	Constant,n=2~8														

This instruction is only applicable to the transistor output type PLC. 8 X ports and a number of Y ports are used to form the matrix input network to expand the channel number for the input signals. Where:

S is the starting address of the hardware port X for the matrix-scan input. It should be a numbered component with a lowest bit of 0 like X0, X10¡, and it occupies 8 continuous bits.

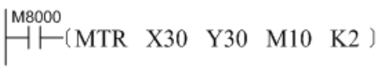
D1 is the starting address of the hardware port Y for the matrix scan input. It should be a numbered component with a lowest bit of 0 like Y0, Y10¡, and it occupies n (n=2 to 8) continuous bits.

D2 is the starting address of the storage unit of the reading status of the matrix scan. It should be a numbered component with a lowest bit of 0 like Y0, Y10, etc.

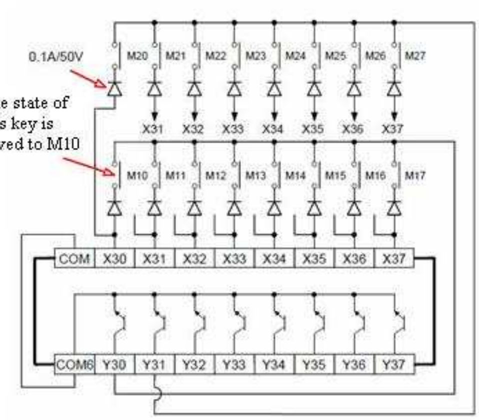
n is the number of the columns of the matrix scan, the number of the Y outputs used by the scan.

Typically, the normally ON contact M8000 is used as the condition contact for this instruction.

Programming example:



The following wiring is applicable:



Considering a response delay of 10ms for the X input filtering. The Y30 and Y31 outputs will be sequentially interrupted for each 20ms to perform the instant input/output process.

Each time after the automatic reading operation is done, the M8029 sign will switch ON a scanning cycle.

A scanning input with a maximum of 64 points can be achieved using 8-point X output and 8-point transistor Y output. But it is not suitable for high speed input operations because it needs a time of 20ms¡Á8 colums = 160ms to read each input. Therefore, the ports after X20 are typically used as the scanning inputs.

This instruction is allowed to be used only once in the program.

MUL instruction

Instruction Description

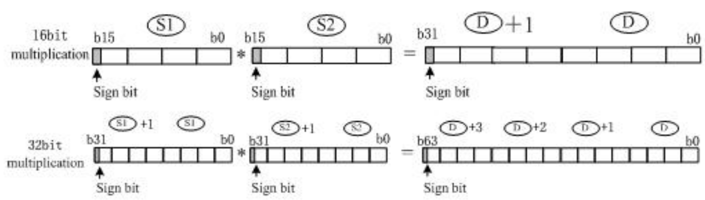
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
MUL	multiplication operation	16	No	MUL S1 S2 D	7
MULP		16	Yes		7
DMUL		32	No		13
DMULP		32	Yes		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔		

A contact drive is required. There are 3 operational variables, the algebra product of the values of S1 and S2 is stored into D . All the variables involved are treated as signed numbers, the upperest bit indicates the sign, 0 for positive numbers and 1 for negative numbers.

The V and Z component in the above table are only applicable for 16bit operations.

For 32bit operations, the variable address in the instruction is the lower than 16bit address, the neighboring higher numbered address unit is 16bit higher to avoid repeating or overwriting. As the calculation result can only be of 32bit, float operation instruction EMUL should be used for operations might exceed the range of 32bit.



Programming example:

When M8 is set, the content of multiplicand D100 is multiplied by the content of multiplier D110 and saved to D120.If D100=K5, D110=K9, then D120=5×9=K45
If D100=K1234, D110=K5678, then D120,d121=1234×5678=K7006652. Note that the product is higher than 16bit in this case and it needs to use neighboring high bits D121 and D120 in D.

NEG instruction

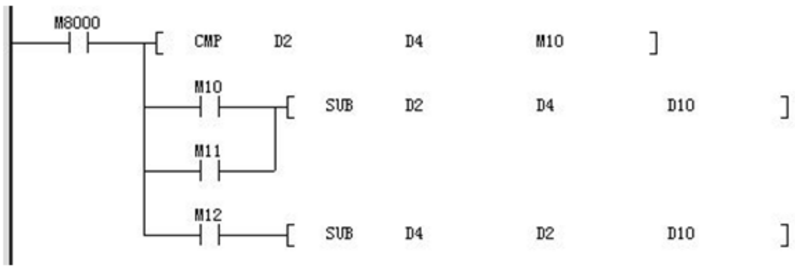
Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
NEG	complementation	16	No	NEG D	3
NEGP		16	Yes		3
DNEG		32	No		5
DNEGP		32	Yes		5

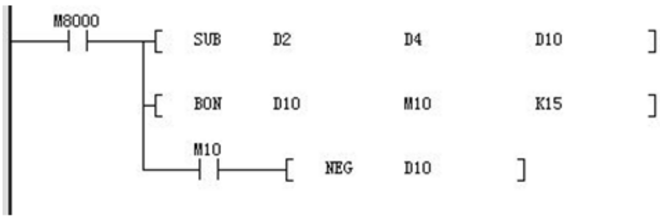
A contact drive is required and there is 1 operational variable. The value of D is conversed bit by bit, and then the resulted value is added up with 1 and stored into D. Typically this instruction is of pulse execution type. To get the absolute value corresponding to the negative BIN value, the NEG instruction can be used.

Programming example:

Take the absolute value of a subtraction operation



If D2>D4, then M10=On; if D2=D4, then M11=On; and if D2<D4, then M12=On. By this way it can be ensured that the value of D10 is always positive. This program can be illustrated by the following process flow



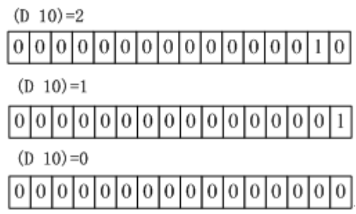
When bit15 of D10 is "1" (indicates that D10 is negative), M10=On. The complemented value of D10 by instruction NEG will be the absolute value of D10.

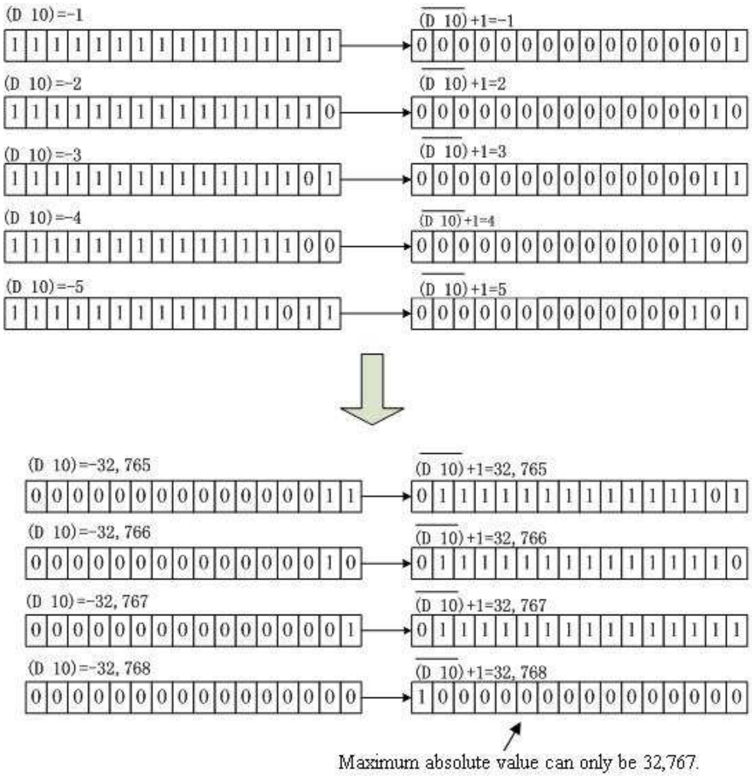
In both examples above, the result for D10 is K4 if D2=K4,D4=K8 or D2=K8,D4=K4.

Additional remarks:

1. If a number is positive or negative, it is indicated by the value of the highest bit (the leftmost), "0" for positive and "1" for negative.

2. If the value of the highest bit is 1, the NEG instruction can be used to convert it into the absolute value.





PID operations

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
PID	PID operations	16	No	PID S1 S2 S3 D	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1													✔		
S2													✔		
S3													✔		
D													✔		

The present instruction is used for implementation of PID operations used to control parameters of close-loop control systems. PID control is widely applied in mechanical equipments,pneumatic equipments, constant-pressure water supply and electronic equipments, etc. Where:

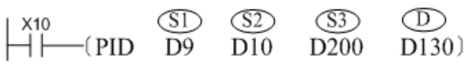
S1 denotes the object value of PIC control;

S2 denotes the feedback value of practical measures;

S3 denotes the starting address of storage buffers preserving parameters set and intermediate results essential for PID operations. It occupies up to 25 variable units of subsequent addresses, and its span is D0~D7975. Furthermore, it is advisable that power-off holding area is assigned and the set values are still preserved after the power is off, or assignments to storage buffers are needed before starting operations for the first time. For more details about functions and parameters description of every unit of S3 you can refer to instructions in the present session

D denotes the storage location of PID calculation results. Please assign D as not the battery holding area. Initialized reset manipulation is needed before starting operations for the first time.

Programming example:



Parameters of D are explained just as follows:

Object values of the PID adjustment are preserved in D9, and D10 preserves the close-loop feedback values. Notice that the uniform dimension must be adopted by D9 and D10. For example, they can both feed the unit of 0.01MPa or 1degree Celsius, etc;

D200~D224:25 units together are applied to preserve the set values and the process values of PID operations, which must be assigned one by one before the first PID calculation.

D130 is then applied to preserve calculated control output values, which is used to control executable moves.

Functions and parameter descriptions of units of S3 are showed as the following table:

unit	Function	Setting instructions
S3	Sampling time(TS)	Setting span is 1~32767(ms), but greater than the scanning period of the PLC routine.
S3+1	Action direction(ACT)	bit0:0 means positive-going action;1 means negative-going action bit1:0 means the alarm of input variable is invalid;1 means the alarm is valid bit2:0 means the alarm of output variable is invalid;1 means the alarm is valid bit3:unavailable bit4:0 means no action of self-setting;1 means action of self-setting bit5:0 means the setting of output high-low limit is invalid;l means the setting is valid bit6 ~ bit15 are unavailable.Besides,please don't set bit5 and bit2 both to be ON at the same time.
S3+2	Input filter constant(α)	0~99[%], 0 means no input filter.
S3+3	Proportional gain(Kp)	1~32767[%]
S3+4	Integration time(T1)	0~32767(×100ms),0 is processed as∞ (no integration).
S3+5	Differentiation gain (KD)	0~100[%],0 means no differentiation gain.
S3+6	Differentiation time(TD)	0~32767(×10ms),0 means no differentiation.
S3+ (7~19)	This unit is occupied by internal processing of PID operations, and should be reset before the first time operation.	
When bit1=1,bit2=1or bit5=1,+(20~24)are occupied defined as follows:		
S3+20	Set alarm value for input variable (increment side)	0~32767,(valid when bit1=1)
S3+21	Set alarm value for input variable (decrement side)	0~32767,(valid when bit1=1)
S3+22	Set alarm value for output variable (increment side)	0~32767,(valid when bit2=1,bit5=0)
		Output high limit -32768~32767 (valid when bit1=1,bit5=1)
S3++23	Set alarm value for output variable (decrement side)	0~32767(valid when bit2=1,bit5=0)
		Output low limit -32768~32767 (valid when bit1=0,bit5=1)
S3+24	Alarm output	bit0:overflow of the input variable (increment side) bit1:overflow of the input variable (decrement side) bit2:overflow of the output variable (increment side) bit3:overflow of the output variable (decrement side)(valid when bit1=1 or bit2=1)

PID's theoretical computational formulas are as follows:

Positive logic

$$\triangle MV = Kp \left| (EVn - \mathbf{EVn\text{-}1}) + \frac{\mathbf{Ts}}{\mathbf{Ti}} EVn + Dn \right|$$

$$\mathbf{EVn = PVnf - SV}$$

$$Dn = \frac{\mathbf{T_D}}{\mathbf{Ts} + \boldsymbol{\alpha}_D * \mathbf{T_D}}(-2PVnf\text{-}1 + PVnf + PVnf\text{-}2) + \frac{\boldsymbol{\alpha}_D * \mathbf{T_D}}{\mathbf{Ts} + \boldsymbol{\alpha}_D * \mathbf{T_D}} * Dn\text{-}1$$

$$MVn = \Sigma \triangle MVn$$

Negative logic

$$\triangle MV = Kp \left| (EVn - \mathbf{EVn\text{-}1}) + \frac{\mathbf{Ts}}{\mathbf{Ti}} EVn + Dn \right|$$

$$\mathbf{EVn = SV - PVnf}$$

$$Dn = \frac{\mathbf{T_D}}{\mathbf{Ts} + \boldsymbol{\alpha}_D * \mathbf{T_D}}(2PVnf\text{-}l - PVnf - PVnf\text{-}2) + \frac{\boldsymbol{\alpha}_D * \mathbf{T_D}}{\mathbf{Ts} + \boldsymbol{\alpha}_D * \mathbf{T_D}} * Dn\text{-}1$$

$$MVn = \Sigma \triangle MVn\triangle$$

EVn :deviation of the present sampling
Dn :the present differentiation
EVn-1:deviation before 1 period
Dn- 1 :differentiation before 1 period
SV :object value
Kp :proportional gain
PVnf :measured value of the present sampling(after filtering)
Ts :sampling period
PVnf-l L: measured value before 1 period(after filtering)
Tl : integration constant

PVnf-2 :measured value before 2 periods(after filtering)
ΔMV: output variable
ΔαD : differentiation gain
ΔMVn :the present operation variable

....

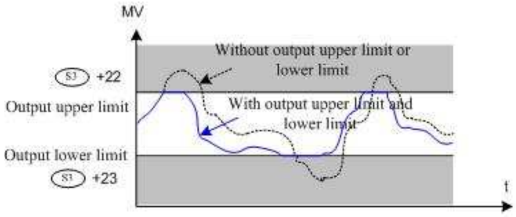
PVnf is represented by calculated value according to read-in measured values applying the working equation below:

[measured value after filtering PVnf] = PVn + L (PVnf - l - PVn)
PVn :measured value of the present sampling
L :filter coefficient
PVnf - 1 : measured value before 1 period (after filtering)

Positive logic is also referred to as positive direction, e.g. heat power adjustment of constant-temperature control system etc., which belongs to positive logic PID control; Negative logic is also referred to as negative direction, e.g. radiator fans' running speed control of constant-temperature control system, which belongs to negative logic PID control.

Settings for output upper limit and lower limit:

For the purpose of achieving the best capacity, the output range of PID adjustment is limited, such as level signal amplitude range and adjustable frequency range of inverter, etc. Thus, output upper limit and lower limit of PID operations should be set according to practical circumstances so that executive equipments of close-loop systems can work under normal running parameters.

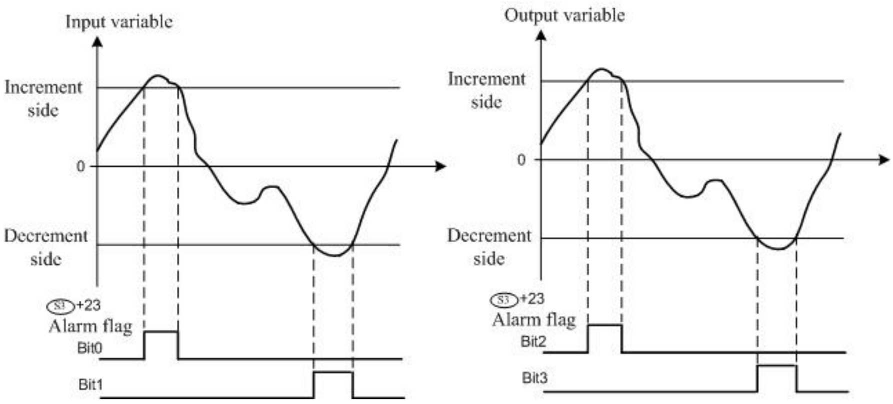


Settings for over-limit alarm:

If we need to check whether the input (feedback) value variation of the controller is over-limit (upper-limit or lower-limit) or not, whether output variation of PID operations is too great (exceeding upper-limit or lower-limit) or not, then we can set bit1 and bit2 in the ACT(S3+1)unit both to be ON, and bring the alarm function into service; besides, through setting alarm limit values of variations respectively in S3+(20~23) units, we can read the over-limit condition of parameters in S3+24 unit when operating. The adjustments said simplify operations when adjusting conditions need to be assessed.

Moreover, when the alarm function of output variation is utilized, it is necessary to make sure that bit5 in S3+1 unit is set OFF.

Here, in the "variation", we've come up with = (the preceding value) - (the present value), and corresponding action schematic diagram is shown as follows:



Therefore, the alarm lower-limit setting value of "decrement side" is negative. To avoid too wide input fluctuation, we generally use some hardware filter processing for input signals or software filter processing in user routines.

Settings for PID constants:

Selecting the following constants: Kp, Ki, Kd, Ts, etc. is critical for PID control performance. The qualitative influence of these parameters for stability is shown as follows:

1)When the proportional gain of Kp is too great, overshoot occurs bringing instability; With Kp too little, adjusting process turns slow; Thus, with a proper setting to Kp response speed of the system PID adjustment can show a good effect.

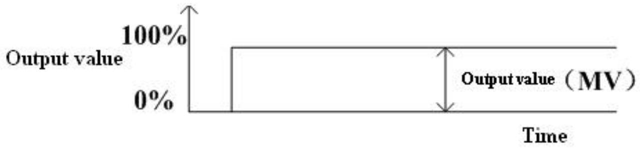
2) When integration time Ti is too large, adjusting process is slow; With Ti is too little, output discontinuity turns obvious, and it's hard to make the system stable; Thus, with a proper setting to Ti the system stability can be reached much faster with a small deviation.

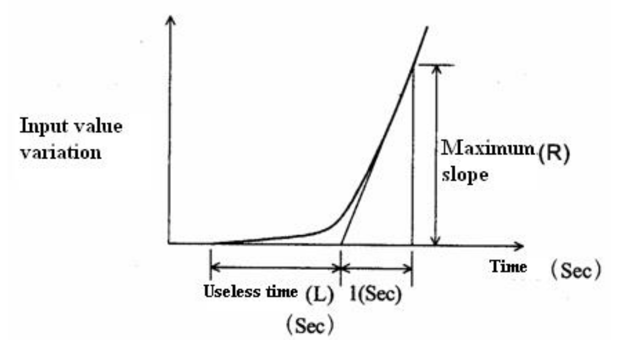
3) When differentiation gain Kd is too large, the system is susceptible to self-oscillation resulting in instability; On the other hand, with a little Kd the influence on stability turns also little; Thus, a proper setting to Kd is beneficial for the system to make a fast approximation to the set value. Furthermore, providing this parameter has a large influence on stability, differentiation adjustment usually is not used in most applications, i.e. Kd is usually set to 0.

4) The sampling period Ts corresponds to time interval of achieving PID operations, and its set value should be greater than executive time of PLC routine scanning. In order to guarantee the PID adjustment effects, it is advisable to run the PLC routines according to a mode of invariable scanning period, or just run PID in the timing interrupt.

Solution methods of the 3 constants of PID

To realize good and expected control results through a PID control, it is indispensable to work out the optimum values suitable to the constants (parameters) of the control system. Thus, it is recommended to work out the optimum of those three parameters, i.e. Kp, Ti, Td. One of the many solution methods is step response explained below. First, a 0--100% (also can be 0--75% or 0--50%) step input signal is imposed to the control system. And then action properties can be asessed with a maximum slope R and useless time L, according to input variation. Hence, three PID constants can be similarily worked out.





<Action features and 3 constants>

	Proportional gain (KP)[%]	Integration time (T1)[×100ms]	Differentiation time (T1)[×100ms]
With only proportional control (p-action)	$\frac{1}{RL}$× Output value (MV)	———	———
With only proportional control (p-action)	$\frac{0.9}{RL}$× Output value (MV)	33L	———
With only proportional control (p-action)	$\frac{1.2}{RL}$× Output value (MV)	20L	50L

Automatic tuning function

Automatic tuning function is used for obtaining the optimum PID control. Significant constants, including action direction (bit0 in S3 + 1), proportional gain(S3+3), integration gain(S3+ 4) and differentiation time(S3+ 6) can be set automatically by means of step response method.

- Automatic tuning method

1 Transfer output values used by automatic tuning to output value D used for self-setting. Please use these values in the range of 50%--100% of possible maximum output values according to the output equipment.

2 Please set the parameters that cannot be self-set (e.g. sampling time, input filter, differentiation gain, etc.) as well as object values etc. Besides, self-setting cannot be correctly achieved if the essentials below aren't satisfied, which should be paid enough attention to.

3 As soon as bit4 in S3+l(ACT) is set as 1, self-setting begins. And when the variation from the measured value at the beginning of self-setting to object value exceeds 1/3, self-setting process ends along with bit4 in S3+l(ACT) turning to 0 automatically.

Important

◦ Setting of object value

After self-setting, if the deviation of measured value and object value is not above 150, it cannot be correctly self-set.Therefore, if the deviation is not above 150, setting of self-setting shall be implemented before setting of object value.

◦ Sampling time must be over 1 second (100ms) during self-setting. Besides, a sampling time which is much longer than output variation period is recommended.

Important

Please start self-setting when the system is in a stable state. Self-setting cannot be started when the system is not in a stable state.

Error code

When something is wrong with the set value of control parameters or data in PID operations, operational error M8067 turns ON. According to the specific error correspondingly, the following data will be stored in D8067:

Code	Error description	Solution conditions	Solution methods
K6705	Operators applying instructions are out of the object software range	PID instructions operations end	Please verify control data
K6706	Operators applying instructions are out of the object software range		
K6730	Sampling time(TS) is out of the object software range(TS<0)		
K6732	Input filter constant is out of the object α range(α<0 or 100≤α)		
K6733	Proportional gain(Kp)is out of the object range(Kp<0)		
K6734	Integration time(Ti)is out of the object range(Ti<0)		
K6735	Differentiation gain(KD)is out of the object range(KD < 0 or 201≤KD)		
K6736	Differentiation time(TD) is out of the object range(TD<0)		
K6740	Sampling time(Ts)¡Üoperational period	PIDinstructions operations continues	
K6742	Variation of measured values is out of the range of PV<-32768 or 32767 < PV		
K6743	Deviation out of the range of EV<-32768 or 32767 < EV		
K6744	Calculated integration value out of the range of - 32768 ~32767		
K6745	Due to differentiation gains out of range, differentiation value is out of the range		
K6746	Calculated differentiation value is out of the range of -32768~ 32767		
K6747	PID operations results out of the range of -32768~32767		
K6750	Self-setting results aren't expected	Self-setting ends	When the variation between measured value and object value is above 150 or measured value exceeds 1/3 of object value at the beginning of self-setting, please repeat the self-setting after finishing with verifying of measured values and object values.
K6751	Self-setting action direction is out of step	Self-setting continues	Actions directions predicted according to the measured values at the beginning of self-setting and practical action directions are inconsistent. Please make sure that the relations among the object value, the output value for self-setting use and the measured value are correct, after which self-setting can be repeated.
K7652	Self-setting action isn't expected	Self-setting	Self-setting cannot do correct actions owing to measured values' thereabout fluctuation. Please set the sampling time to be far greater than output variation period, and increase the input filter constant. Then please do the self-setting one more time after altering those settings.

> Important
>
> The correct measured value must be read into the PID measured values (PV) before executing PID operation. Specially, pay attention to the conversion time when PID operation for input value of analog input modules is implemented.

Programming notes:

1) PID instructions can be used many times in routines, and can be executed at the same time. However, those S3 variable areas used in PID instructions must not overlap with each other; Besides, PID instructions can also be used in step instructions, jump instructions, timing interrupts, or subroutines. Under these circumstances S3+7 storage buffer need to be reset before executing PID instructions.

2) The maximum error of sampling time Ts is -(1 operation period+ l ms)~ +(1 operation period). If sampling time Ts ≤ 1 operation period of the programmable controller, then the following PID operation error (K6740) occurs and PID operation is executed with Ts=operation period. Under this circumstance, it is advisable to use the constant scanning mode or use PID instructions in timer interrupts (16~18).

PLSR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
PLSR	Pulse output with acceleration/deceleration output	16	No	PLSR S1 S2 S3 D	7
DPLSR		32	No		17

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S3					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D		✔													

Since relays are not suitable for high frequency actions, only the transistor output type PLC is suitable to use the present instruction. The present function is represented by the pulse output instruction for fixed size transmission with acceleration/deceleration function. Where:

S1 represents the maximum frequency of the out pulse set with a range from 10~100,000Hz;

S2 represents the number of the output pulses set. The setting range is 110~32,767 for 16bit instructions and 110~2,147,483,647 for 32bit instructions. If the number of pulses set is less than 110, the pulses can't be output normally;

S3 represents the acceleration/deceleration time set with a range from 50~5000 (ms). The deceleration time is the same as the acceleration and both are measured in ms. It should be noted during the setting that: (In the new version of H2U series PLC deceleration time can be specified separately, refer to the introduction later)

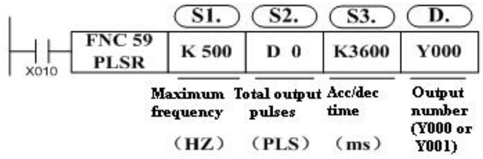
D represents the pulse output port, for 3624MT/2416MT model only Y0 or Y1 can be specified, Y0/Y1/Y2 for 1616MT/3232MT model, and Y0/Y1/Y2/Y3/Y4 for MTQ model. This port can't be repeated with the output port of the PLSY instruction.

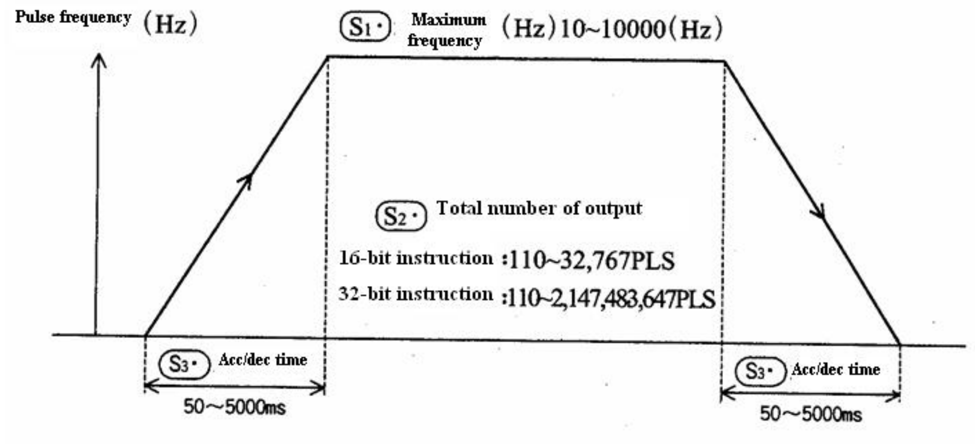
Instruction for use:

- The instruction is executed in an interruption way, therefore it will not be influenced by the scanning cycle;
- When the instruction power flow is OFF, the deceleration stop is active; when the power flow is changed from OFF¡úON, the pulse output process starts over again;
- During the pulse output process, changing of the operand will not affect the current output. The changed items will be effective the next time the instruction is executed. The M8029 sign is switched ON when the execution of the instruction is completed;
- The 40-point MT model PLC and 60-point MT model PLC can use 2 PLSR instructions and 2 PLSY instructions at the same time, corresponding to the Y0 and Y1 ports respectively. For MT model PLC with other number of points, 3 PLSR instructions or 3 PLSY instructions can be used at the same time, corresponding to the Y0, Y1, Y2, Y3 and Y4 ports. For detailed specifications refer to the hardware user's guide for programmable controller.
- The process can't be repeated with the output port number of the PWM instruction;

When starting the PLSR instruction again, a delay of 1 scanning cycle is required after the last pulse output operation is finished (M8147=0 when Y0 finished; M8148=0 when Y1 is finished; M8149=0 when Y2 is finished; M8150=0 when Y3 is finished; and M8151=0 when Y4 is finished) before the restarting (In the new version of H2U series PLC through proper configuration the limitation can be bypassed, for details please refer to the instructions in section 8.7 of the appendix);

Programming example:





The special registers corresponding to each output port are listed as follow:

Register		Definition	Remarks
D8140	Lower byte	Number of total pulses output to Y0 port set in the PLSY or PLSR instruction	Applicable instructions: use DMOV K0 D81xx to perform clear operation
D8141	Upper byte		
D8142	Lower byte	Number of total pulses output to Y1 port set in the PLSY or PLSR instruction	
D8143	Upper byte		
D8150	Lower byte	Number of total pulses output to Y2 port set in the PLSY or PLSR instruction (3624MT/2416MT doesn't feature this port)	
D8151	Upper byte		
D8152	Lower byte	Number of total pulses output to Y3 port set in the PLSY or PLSR instruction (Only applicable to MTQ)	
D8153	Upper byte		
D8154	Lower byte	Number of total pulses output to Y4 port set in the PLSY or PLSR instruction (Only applicable to MTQ)	
D8155	Upper byte		
D8136	Lower byte	Accumulative value of the number of the pulses already output to Y0 and Y1	
D8137	Upper byte		

The output frequency range of this instruction is from 10~100,000Hz. When the high speed conversion with the maximum or accelerated/decelerated speed exceeds the range, it will be converted (ascended or descended) to a value in this range automatically before it is executed. However, the minimum frequency can actually be output is determined by the following formula:

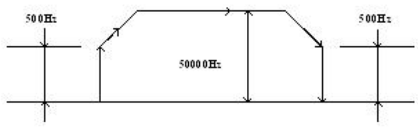
$$\sqrt{\text{Max frequency (S1) Hz} \div (2\times(\text{Acceleration/deceleration time (S3) ms} \div 1000))} = \text{Min number of pulse output frequency}$$

The frequencies at the early stage of acceleration and at the late stage of deceleration must not be lower than the above calculation result.

[Example] Maximum speed:, Acceleration/deceleration time

$$\sqrt{50000 \div (2\times(100\div 1000))} = 500\text{Hz}$$

When maximum frequency S1 is specified to 50000Hz, the actual output frequency at the early stage of acceleration and at the late stage of deceleration is 500Hz.



Note:

In the new version of H2U series PLC the PLSR, DRVI and DRVA instruction, etc. have enhanced functions;

1. It is possible to modify (larger or smaller) the number of the pulses during the operation through switching ON the special bits M8135~M8139 (corresponding to Y0~Y4 respectively). Meanwhile, the deceleration time is defined by the following registers respectively:

Port no.	Special bits	Register for modifying deceleration time
Y000	M8135	D8165
Y001	M8136	D8166
Y002	M8137	D8167
Y003	M8138	D8168
Y004	M8139	D8169

2. Through switching ON the special bits M8085~M8089 (corresponding to Y0~Y4 respectively) the following functions can be achieved: the next pulse output instruction can be started immediately if the special drive bit is ON, without the processing of the last void power flow;

3. Through switching ON the special bits M8090~M8094 (corresponding to Y0~Y4 respectively) the following functions can be achieved: it is possible to interrupt the pulse output, as detailed below:

Port no.	Special bits	Related user interrupt
Y000	M8090	I502
Y001	M8091	I503
Y002	M8092	I504
Y003	M8093	I505
Y004	M8094	I506

PLSV instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
PLSV	Variable speed pulse output	16	No	PLSV S D1 D2	9
DPLSV		32	No		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D1		✔													
D2		✔	✔	✔											

The present instruction outputs pulse frequency according to the specified port, frequency and operation direction without acceleration/deceleration. The pulse output will be stopped directly when the driving power flow is ineffective. Only the PLC with the transistor output can execute the instruction. Where:

S represents the specified output pulse frequency with a range from 1~32,767, -1~-32,768 Hz for 16bit instructions, and 1~100,000Hz,-1~-100,000Hz for 32bit instructions. Where the negative sign indicates instruction signals in inverse operation;

D1 represents the pulse output port. Only Y0 or Y1 can be specified for the 3624MT/2416MT model, and Y0/Y1/Y2 for 1616MT/3232MT mode, while Y0/Y1/Y2/Y3/Y4 is applicable for the MTQ mode;

D2 is the operating direction output port or variant. When the output is in ON state, the system is operating in the forward direction, and vice versa.

Programming example:

```
        S       D1   D2
M1
─┤├──[PLSV  K10000   Y1   Y4]
```

Expressing using sentences, when M1 is ON, the Y1 port will output pulses of 10 kHz frequency and the Y4 port will be used to control the motion direction. Y4=ON indicates the positive direction.

The involved system variables during the execution of the present instruction:

1. D8141 (the upper byte), D8140 (the lower byte): Y000 represents the number of the output pulses decreasing when reversing. (using 32bit)
2. D8143 (the upper byte), D8142 (the lower byte): Y001 represents the number of the output pulses decreasing when reversing. (using 32bit)
3. D8151 (the upper byte), D8150 (the lower byte): Y002 represents the number of the output pulses decreasing when reversing. (using 32bit)
4. D8153 (the upper byte), D8152 (the lower byte): Y003 represents the number of the output pulses decreasing when reversing. (using 32bit)
5. D8155 (the upper byte), D8154 (the lower byte): Y004 represents the number of the output pulses decreasing when reversing. (using 32bit)
6. M8145:Y000 represents the pulse output stopped (instantly)
7. M8146:Y001 represents the pulse output stopped (instantly)
8. M8152:Y002 represents the pulse output stopped (instantly)
9. M8153:Y003 represents the pulse output stopped (instantly)

10. M8154:Y004 represents the pulse output stopped (instantly)
11. M8147:Y000 represents monitoring during the pulse output process (BUSY/READY)
12. M8148:Y001 represents monitoring during the pulse output process (BUSY/READY)
13. M8149:Y002 represents monitoring during the pulse output process (BUSY/READY)
14. M8150:Y003 represents monitoring during the pulse output process (BUSY/READY)
15. M8151:Y004 represents monitoring during the pulse output process (BUSY/READY)

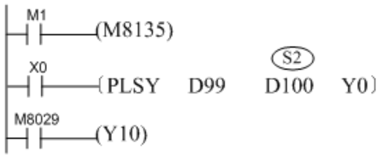
PLSY instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
PLSY	Pulse output	16	No	PLSY S1 S2 D	7
DPLSY		32	No		13

1). The following functions can be achieved by switching ON the special bits M8135~M8139 (corresponding to Y0~Y4 respectively): the number of pulses can be increased or decreased during operation.

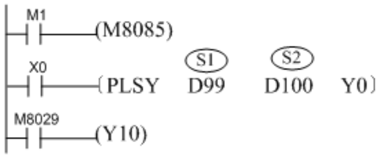
Programming example:



This special function is enabled when M1 is set to ON. X0 is turned on to execute PLSY instruction and D100 can be changed during pulse transmission to increase or decrease the number of pulse (notice: the modified content in the D100 should be larger than the number of pulse sent by the instruction.)

2£©. The following functions can be achieved by switching ON the special bits M8085~M8089 (corresponding to Y0~Y4 respectively): when the drive special bit is ON, the next pulse output instruction can be started instantly without the processing of the last void energy flow.

Programming example:



If X0=1, if M8085 is set, the PLSY instruction will restart and start to transmit pulses again according to the frequency and the number of frequency S1, S2(i.e. the current values of D99 and D100), during the pulse transmission process or after the pulse transmission is done.

3£©.The following functions can be achieved by turning ON the special bits M8090~M8094 (corresponding to Y0~Y4 respectively): can execute one user interrupt task after pulse output in the following way:

Port no.	Special bits	Related user interrupt
Y000	M8090	I502
Y001	M8091	I503
Y002	M8092	I504
Y003	M8093	I505
Y004	M8094	I506

If the special function bits M8090~M8094 turn ON, user interrupt task I502¡«I506 will be executed immediately after sending a specified number of pulses.

PRUN instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step

PRUN	Transmission of octal bits	16	No	PRUN Ⓢ Ⓓ	5
PRUNP		16	Yes		5
DPRUN		32	No		9
DPRUNP		32	Yes		9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
Ⓢ							✔		✔						
Ⓓ								✔	✔						

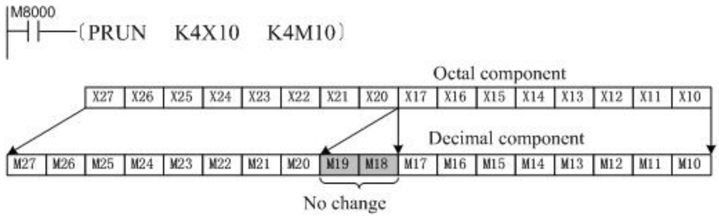
The present instruction is used to copy the bit variables (the width unit is of octal) of the continuous addresses starting with Ⓢ to the bit variable set starting with Ⓓ in batch. Where:

Ⓢ is the starting address of the bit variables to be copied, where the unit digit of the addresses must be 0, such as X10, M20, etc.

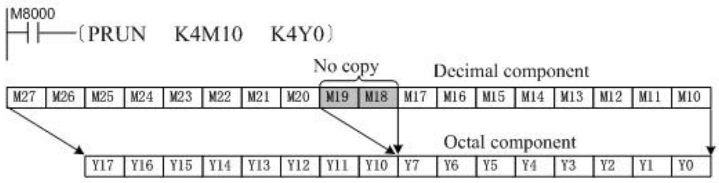
Ⓓ is the starting address of the target bit variables. Also, the unit digit of the addresses must be 0, such as M30, Y10, etc.

Programming example:

Instruction example 1:



Instruction example 2:



PR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
PR	ASCII code printing	16	No	PR Ⓢ Ⓓ	5

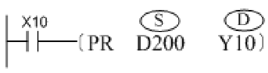
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
Ⓢ											✔	✔	✔		
Ⓓ		✔													

This instruction is used to output the values of the specified variable units byte by byte synchronously through the Y output port. Where:

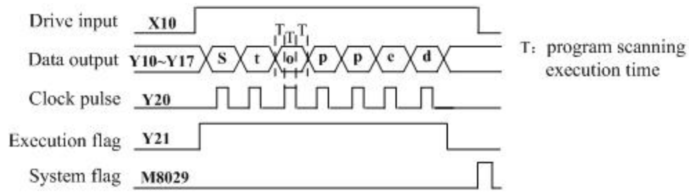
Ⓢ is the starting address of the variable units to be output;

Ⓓ is the starting number of the Y port for output print.

Programming example:



If the ASCII code in D200~D203 is "Stopped", then the corresponding output port signal and their time sequence is as follow:



Instruction for use:

The transistor output type PLC must be used to achieve the function of this instruction;

- During the printing, the printing output will be interrupted when the drive signal X10 is changed to OFF. The printing action will start again when X10 is ON;
- If M8027=OFF, the serial output for a maximum of 8 chars is possible, and 1~16 chars when M8027=ON.
- During the printing output the operation will be ended automatically when chars like "00" occur, and the subsequent text will be skipped. The finish sign M8029 will only switch ON a scanning cycle after the drive energy flow signal is ineffective;

The instruction is executed according to the scanning cycles (T in the diagram). Please use the fixed scanning mode for a short scanning period, and it can be executed in a time definite interruption program for a longer scanning period.

PWM instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
PWM	Pulse band modulation	16	No	PWM S1 S2 D	7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D		✔													

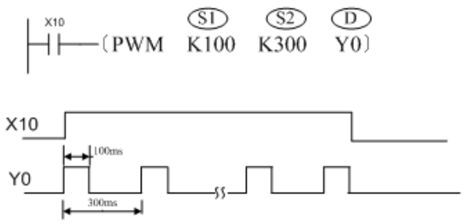
Since relays are not suitable for high frequency actions, only the transistor output type PLC is suitable to use the present instruction. This instruction outputs pulses continuously with the pulse band specified by S1, the pulse period specified by S2 and the port specified by D Where:

S1 is the specified output pulse band, it must meet the requirement S1 = S2. The setting range is between 0~32,767ms;

S2 is the specified output pulse period, it must meet the requirement S1 = S2. The setting range is between 0~32,767ms;

D is the pulse output port. Only Y0 or Y1 can be specified for the 3624MT/2416MT model; Y0/Y1/Y2 can be specified for other MT models; and Y0/Y1/Y2/Y3/Y4 can be specified for the MTQ model, where it must not be repeated with the output ports of PLSY and PLSR instruction. The instruction is executed in an interrupted way, the output is stopped when the instruction power flow is OFF. S1 and S2 can be modified when the PWM instruction is being executed.

Programming example:



RAD instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
	Operation of				

DRAD	angle to radians in binary floating point number	32	No	RAD S D	9
DRADP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

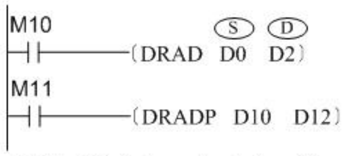
The present instruction is used for binary floating point operation of angle to radians. The formula is [angle unit=radian unit×∏/180], where,

S represents the angle variable quantity which is to store the angle in radians of requested two binary floating-point number.

D is the storage unit of the calculation result.

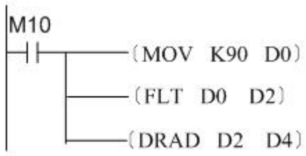
Programming example:

Instruction example 1:



When M10 is ON, implement angle to radian operation for binary floating value in (DO, D1) and save to (D3, D2).

Instruction example 2 :



When M10 changes from OFF to ON, D0 is set to 90 and converted from integral to floating, and the result is saved to (D3,D2). After (D3,D2) is implemented with angle-to-radian calculation, the result, which is π/2(1.570796), is saved to (D5,D4).

RAMP instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
RAMP	Slope signal	16	No	RAMP S1 S2 D n	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1													✔		
S2													✔		
D													✔		
n	Constant,1~32767														

This function of command is carrying on linear interpolation among two given data or appointed time sector in order to output procedure value according to the turn of scanning execution time, until sector terminal endpoint. Where:

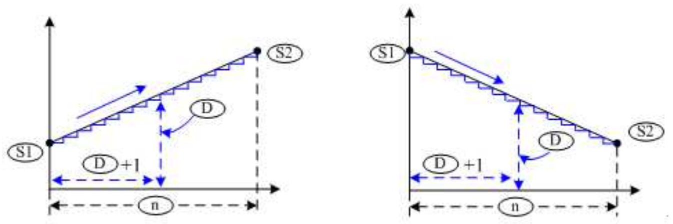
S1 The starting value unit of slope signal

S2 The end-point value unit of slope signal

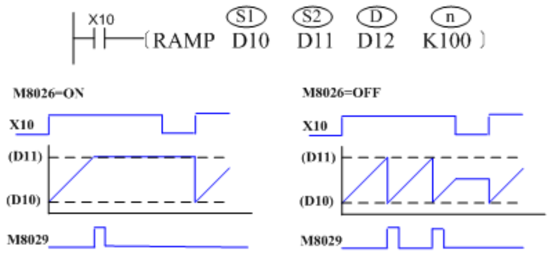
D The memory point for procedure value of linear interpolation signal , yet the timer which is used to count the times of interpolation is stored in unit D+1.

n The times of program scanning execution for process of interpolation .Because the output of interpolation is carried on during main loop, it's necessary to set the program execution to fixed scanning mode .(the demonstration is on M8039 ,D8039)

The interpolation calculation is based on integer number and has discarded the computation decimal. Command function is showed in the chart followed:



There are 2 modes for RAMP command execution which is selected by M8026 sign; After every interpolation, M8029 set a scanning cycle .The execution features is showed in the follow example:



RCL instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
RCL	Instruction to make of 16-bit or 32-bit data left-shift with carry	16	No	RCL D n	5
RCLP		16	Yes		5
DRCL		32	No		9
DRCLP		32	Yes		9

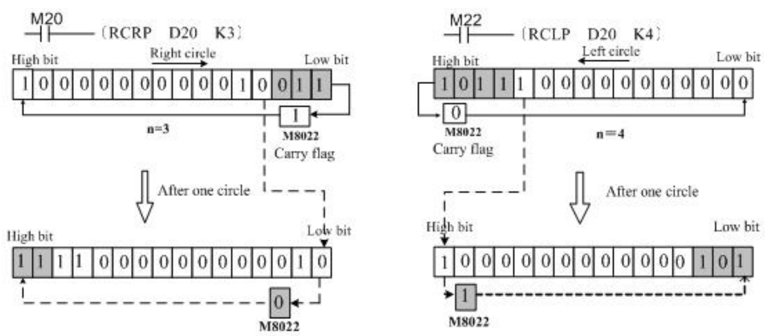
Cycle the content of D and carry mark shifted n bits to left. The instruction usually uses pulse operation type instruction.

When it is 32 bit order, rigister varialbe occupies two units neighbor address behind.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
n	Constant,n=1~16(16bit);n=1~32(32 bit)														

If KnY, KnM or KnS is appointed in D, only K4 (16bit) and K8(32bit) are valid.

Programming example:



RCR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
RCR	Instruction to make of 16-bit or 32-bit data right-shift with carry	16	No	RCR (D) (n)	5
RCRP		16	Yes		5
DRCR		32	No		9
DRCRP		32	Yes		9

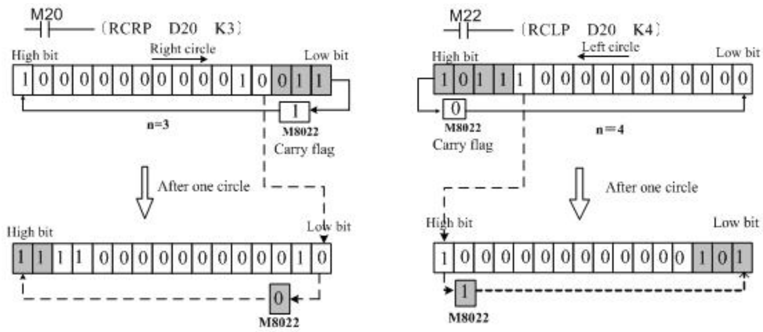
Cycle the content of (D) and carry mark shifted (n) bits to right. The instruction usually uses pulse operation type instruction.

When it is 32 bit order, rigister varialbe occupies two units neighbor address behind.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(D)					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
(n)	Constant;n=1~16(16bit);n=1~32(32 bit)														

If KnY, KnM or KnS is appointed in (D), only K4 (16bit) and K8(32bit) are valid.

Programming example:



RD3A instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
RD3A	Read from analog modules	16	No	RD3A (m1)(m2)(D)	7
RD3AP		16	Yes		7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(m1)					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
(m2)					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
(D)								✔	✔	✔	✔	✔	✔	✔	✔

This instruction provides read of analog input value as the same to Mitsubishi FX0N ~ 3A-type.

(m1) is number of special module; (K0~K7).

(m2) is the channel of analog input; (K1~K2)

(D) is the address of storing readout value.

Programming example

The instruction FROM also can be used to read FX0N~1.21ha.

REFF instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
REFF	Input refreshing (With filter setting)	16	No	REFF n	3
REFFP		16	Yes		3

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
n	Constant,n=0~60,Unit:ms														

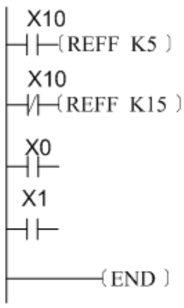
Setting the filter time constants of the X0~X7's inputs to n.

The inputs X0~X7 use digital filter in PLC. The default filter time constants is set by D8020. You can change the value of D8020 to 0~60ms by instruction REFF. The other X ports only have RC filter in hardware. The filter time constant is about to 10ms and its modification is forbidden.

The relative ports' filter time is set to the shortest time automatically when you use the high-speed counter or the interrupting function of the input X.

You can also use MOV to assign a new value to the filter time.

Programming example:



Setting the input filter time of X0~X7 to 5ms as the state of X10 is ON, Setting the input filter time of X0~X7 to 15ms as the state of X10 is OFF.

REF instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
REF	I/0 refreshing	16	No	REF D n	5
REFP		16	Yes		5

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D	✔	✔													
n	constant,n=8~256,it must be multiple of 8														

Update these D components which begin from the address of n immediately.

According to the property of the PLC that it accesses the ports by byte, here is some requirements:

The address of D shoud be the component that the lowest bit is 0 such as X0,X10,...Y0,Y10... etc..

The value of n must be multiple of 8.

Normally, the reading of the state of I/O is ahead of program scan each time. The refreshing of the state of output Y is batch processing after scanning over the program (to End) each time, so the I/O process has delay. If need the latest input information and hope to output the calculate result immediately, you can use the instruction REF to refresh immediately.

¡ñIt can be used in the instruction FOR~NEXT,CJ, etc.

¡ñIt can be used to refresh the I/O to get the latest input information and output the calculate result in intrupt subroutine immediately.

¡ñThe real delay of the state change of I/O is up to the filter time of input components, X0~X7 have the function of digital filter. The time of filtering can be set in the range of 0~60ms (FNC51(instruction:REFF)). The other I/O ports are hardware filter, the filter time is about 10ms. Please refer to the details in the user manual of PLC.

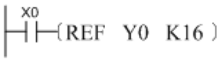
¡ñThe real delay of the state change of I/O is up to the response time of output conponents such as Relay. The output junction in the output refreshing will act after the response time of Output relay(Transistor). The response delay of Output relay is about 10ms(limit to20ms),The hi-speed output of Output transistor is about 10ms, general point output is ablout 0.5ms. Please refer to the details in the user manual of PLC.

Programming example:

Example 1 for instruction:

When processing above program, if the state of X20 is ON, it will read the state of input X0~X17 immediately, updating the input information and there is no input delay.

Example 2 for instruction:



When processing above program, if the state of X0 is ON, it will refresh the state of X0~X17 immediately and update the output signal , there is no necessary to wait for the END.

ROL instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ROL	The instruction to make 16-bit or 32-bit data shift left	16	No	ROL D n	5
ROLP		16	Yes		5
DROL		32	No		9
DROLP		32	Yes		9

Cycle the content of D shifted n bits to left.

The instruction usually uses pulse operation type instruction.

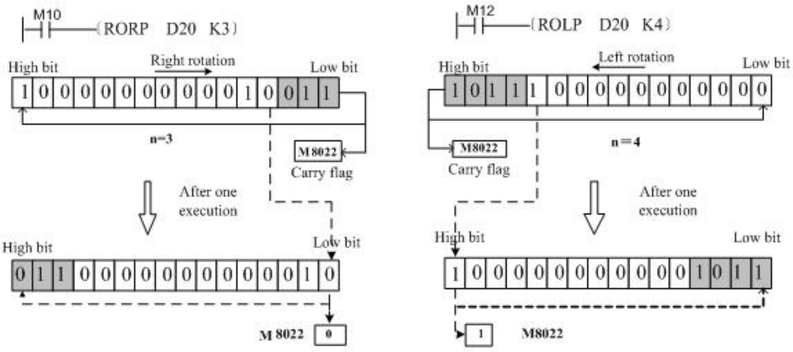
When it is 32 bit order, rigister varialbe occupies two units neighbor address behind.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
n	Constant,n=1~16(16bit);n=1~32(32 bit)														

If KnY ,KnM ,KnS are specified in D,only K4(16bit)and K8(32 bit)valid;

The final bit is circular movement into carry mark.

Programming example:



ROR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ROR	The instruction to make 16-bit or 32-bit data shift right.	16	No	ROR D n	5
RORP		16	Yes		5
DROR		32	No		9
DRORP		32	Yes		9

Cycle the content of D and carry mark shifted n bits to right.

The instruction usually uses pulse operation type instruction.

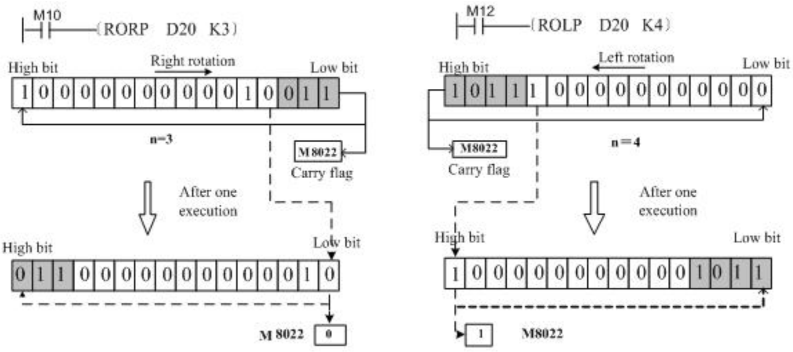
When it is 32 bit order, rigister varialbe occupies two units neighbor address behind.

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
n	Constant;n=1~16(16bit);n=1~32(32 bit)														

If KnY, KnM and KnS are appointed in D , only K4 (16bit) and K8(32bit) are valid.

The final bit is circular movement into carry mark.

Programming example:



ROTC instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ROTC	Rotary workbench control	16	No	ROTC S D m1 m2	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
D		✔	✔	✔											

m1	2~32767,m1≧m2
m2	0~32767, m1≧m2

This instruction is the compact instruction being used to fetch the workpiece on rotary workbench.The position detection signal of rotary workbench shall be configured by desired method to work properly. Where:

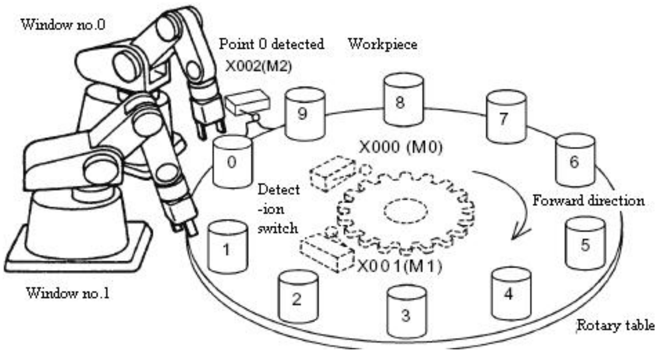
S The initial cell of count variable.

m1 Numbers of station on rotary workbench, which must be m1 ≥ m2;

m2 numbers of low-speed rotary workbench, which must be m1 ≥ m2;

D is the initial cell to storage position detection signal of rotary workbench, which occupies the next 8 bit variable units

Signal configuration as shown below, X0, X1 in the figure connected with the A and B phase output of AB Quadrature Encoder respectively, and we can get the Quadrature signals by mechanical switch. X2 will be used as the detection input of No.0 station ("ON" when turning to No.0 station), the rotational speed, direction, and workstation can be detected by these three signals.



Programming example:

X10 —(ROTC S D200 m1 K10 m2 K2 D M0)

The code actually uses the variable space as follows:

Variable	Function definition	Operation instruction
D200	Used as a counter register	These three units are pre-configured by user program
D201	Call window number setting	
D202	Call workpiece number setting	
M0	A-phase signal	Before scanning of this statement, user program will execute: X0 —(M0) X1 —(M1) X2 —(M2)
M1	B-phase signal	
M2	Point 0 detection signal	
M3	High speed forward direction	If X10=ON, result of M3~M7 can be obtained automatically, If X10=OFF, M3~M7 are all OFF;
M4	Low speed forward direction	
M5	Stop	
M6	Low speed reversed direction	
M7	High speed reversed direction	

In the following user program，let M3~M7 go out from the Y output to control the external executive components. You can use the instruction ROTC only once in the program

RS instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
RS	Serial data transfer	16	No	RS S m D n	9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z

S													✔		
m					✔	✔							✔		
D													✔		
n					✔	✔							✔		

This instruction is a communication transceiver instruction. It sends the data in specified register to the serial ports automatically, and deposit the data to the designated area. This is equivalent to that the user program accesses the communication buffer directly, deals with the communication receiving and sending buffer with the user program, and achieves a custom communication protocol. Where:

S is the initial address of the register area where the data to be sent will be stored;

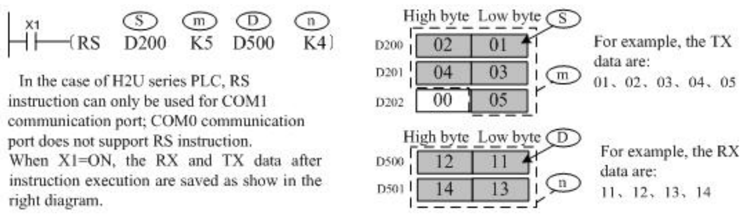
m is the length of the data to be sent (bytes), ranges (0~256);

D is the initial address of the storage register which receives communication data;

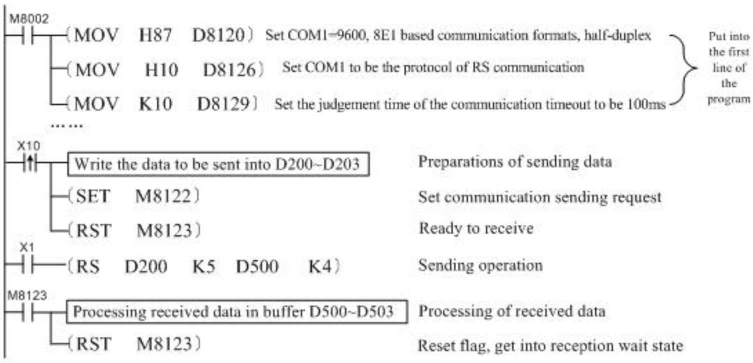
n is the length of the communication data received (bytes), ranges (0~256).

H2U series extend function of the instruction RS. Instruction RS can realize the function of the instruction MOD when choosing MODBUS master protocol. This feature will be described later in this instruction.

Programming example:



You need do some configuration and preparation in serial communication during the actual programming. For example, setting serial port transceiver mode, baud rate, bits, parity bit, setting the software protocol, judging the conditions of overtime, the preparation of sending and receiving buffer data, sending and receiving symbol processing and so on. Then the system can communicate as you expect. Take the last statement for example, a more complete RS communication setup as followes:



SEGD instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SEGD	Seven segment decoder	16	No	SEGD S D	5
SEGDP		16	Yes		5

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

This instruction is to translate the low 4 bites of the data source into 7 segment display code, and store it into the low 8 bits of the destination variable.

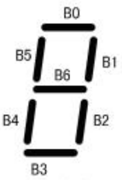
S is the data source waiting to be translated (take the lowest 4 bits b0~b3 of BIN content);

(D) is the variable depositing the 7 yards after decoding.

Programming example:

```
X20      (S)   (D)
─┤├──( SEGD D0  K2Y10 )
```

Operations, when X20 is in the state of ON, translate the low 4 bits of the data storing in D0, and output to port Y10~Y17. The corresponding table used for translating, as follows. The users do not need to prepare the table, the PLC system has already had the check list.

Data		Nixie tube combination	nternal decoding table Value								Decoded symbol
HEX	BIN		B7	B6	B5	B4	B3	B2	B1	B0	
0	0000		0	0	1	1	1	1	1	1	0
1	0001		0	0	0	0	0	1	1	0	1
2	0010		0	1	0	1	1	0	1	1	2
3	0011		0	1	0	0	1	1	1	1	3
4	0100		0	1	1	0	0	1	1	0	4
5	0101		0	1	1	0	1	1	0	1	5
6	0110		0	1	1	1	1	1	0	1	6
7	0111		0	0	0	0	0	1	1	1	7
8	1000		0	1	1	1	1	1	1	1	8
9	1001		0	1	1	0	1	1	1	1	9
A	1010	Each bit	0	1	1	1	1	0	1	1	A
B	1011	corresponds to one	0	1	1	1	1	1	0	0	b
C	1100	stroke segment	0	1	1	1	1	0	0	1	C
D	1101	1=stroke lighted	0	1	0	1	1	1	1	0	d
E	1110	0=stroke extinguished	0	1	1	1	1	0	0	1	E
F	1111		0	1	1	1	0	0	0	1	F

SEGL instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SEGL	Display of seven-segment code	16	No	SEGL (S) (D) (n)	7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(S)					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
(D)		✔													
(n)	constant,n=0~7														

This instruction uses 8 or 12 Y ports for the display driver of the 4 or 8-bit seven-segment digital tube latch, the display mode is the scan driver mode. Where:

(S) is the data to be displayed, its value will be sent to the digital tube for display after BCD conversion.

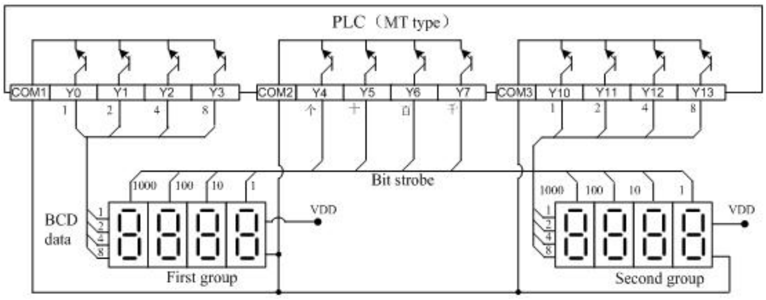
(D) is the start address number of the port used for display driver;

(n)is the settings related of the data show's group number¡¢signal's positive and negative logic and etc..You can see the following detailed description.

Programming example:

```
X10      (S)  (D)  (n)
─┤├──( SEGL D0   Y0   K6 )
```

Corresponding hardware connection is as follows. The contents of D0 are displayed in the first group of digital tube, the contents of D1 are displayed in the second group of digital tube and the procedure operation will run error when D0 or D1's numerical reading exceeds 9999:



The digital tubes in the wiring diagram come with the data show's latch, decoding and driving of 7 segment digital tube, negative logic type (the input data is considered as 1, or strobe when input port is low) 7-segment digital display tubes. In the display processing, PLC's Y4 ~ Y7 port will automatically scan cycle and only one port is ON and as a bit strobe. In this moment, the data of Y0~Y3 port is the BCD code data sent to the corresponding bits and when bit strobe signal change from the ON → OFF, it will be latched to the latch of digital tube. The digital tubes will display the number after internal decoding and driving . PLC systems will deal with Y4 ~ Y7 cycle in turn and by the same process until all the 4 bits has been processed. Similarly, Y10 ~ Y13 is the second group data output port of 4-bit digital tubes and share Y4 ~ Y7 bit strobe line, so the process is in the same and both groups' display is processed at the same time. For the example, the first group will display 2468 and the second group will display 9753 when D0=K2468,D1=K9753.

12 scan cycle is necessary to refresh one display. The flag of M8029 is set to ON after that and it need one scan cycle. The choice of (n) : according to the effects of positive and negative logic of PCL,7 segment code and so on, it can be select by following principle:

If there is one group has 4 digits, n=0~3.If there are two groups have 4 digits, n=4~7.

Group number displayed	Group 1				Group 2			
Polarity of Y data output	PNP		NPN		PNP		NPN	
Strobe and data polarity	Same	Opposite	Same	Opposite	Same	Opposite	Same	Opposite
Value of (n)	0	1	2	3	4	5	6	7

When the polarity of PLC's transistor output and the input polarity of 7 segment display is equal or not, it can match by the set-value of n.

The polarity of output Y in H2U series transistor output type is NPN. This instruction only can be used twice.

Instruction for use:

Because Relay is not suit of higher frequency scanning output, it can be used of PLC which type is Output transistor.

SER instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SER	Data search	16	No	SER (S1) (S2) (D) (n)	7
SERP		16	Yes		7
DSER		32	No		17
DSERP		32	Yes		17

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(S1)							✔	✔	✔	✔	✔	✔	✔	✔	✔
(S2)					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
(D)							✔	✔	✔	✔	✔	✔	✔	✔	✔
(n)					✔	✔							✔		
	Value range: 16bit instruction: n=1~256; 32bit instruction:n=1~128														

The instruction is to search the unit(s) with same data, maximum value and minimum value.

(S1) is the starting address of the data array:

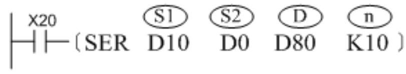
(S2) is the data, which is to be searched;

(D) is the starting address of storage range for search result;

(n) is the length of data range, which is to be searched.

When using 32bit instruction, S1 S2 D are all pointing to 32bit variable, n is also calculated according to 32bit variable width.

Programming example:



S1	Example of data to be retrieved	S2	Components sn	Parameter status
D 10	(D 10)=K100	Comparative data (D0) =K100	0	Equal
D 11	(D 11)=K123		1	
D 12	(D 12)=K100		2	Equal
D 13	(D 13)=K98		3	
D 14	(D 14)=K111		4	
D 15	(D 15)=K66		5	Minimum
D 16	(D 16)=K100		6	Equal
D 17	(D 17)=K100		7	Equal
D 18	(D 18)=K210		8	Maximum
D 19	(D 19)=K88		9	

Search result

D	Parameter	Definition
D80	4	Number of equal paremeters
D81	0	Serial number of the first equal parameter
D82	7	Serial number of the last equal parameter
D83	5	Serial number of the minimum parameter
D84	8	Serial number of the maximum parameter

Instruction for use:

- When instruction power flow X20 is ON, the operation is implemented;
- The comparison method is signed algebra comparison, for example -8<2;
- When there are several minimum or maximum, the components with the largest serials number are displayed respectively;

The storage units for search results occupy five continue units started with D. If there is no same data, D80~D82 in above example are all 0.

SFRD instruction

Instruction Description

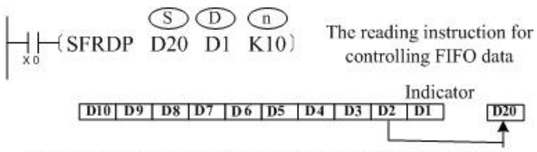
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SFRD	Shift read (the reading instruction for controlling FIFO data)	16	No	SFRD S D n	7
SFRDP		16	Yes		7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔	✔	✔	✔	✔	✔	✔		
D								✔	✔	✔	✔	✔	✔		
n	Value range:16bit instruction:n=1~256; 32bit instruction: n=1~128														

Read the first itme in "FIFO" arrary S to D , and then implementing right shift one word for array S with degressive array point.The first numbered device is taken as point. When implementing instruction, the point content is subtracted by 1, and then the device value specified by S will be written to FIFO D data tandem location specified by point.If the point is 0, the instruction will not be processed according to above operation, and 0 flag M8020 will be set to 1.

The instruction usually uses pulse operation type instruction.

Instruction for use:



When X0 is changed from ON to OFF, the instruction will act as follows. (D10's content will remain unchanged),
1: D2's content will be read out and transmitted to D20.
2: D10~D3 will be right shift by one register.
3: Indicator D1's content will be decremented by 1.

SFTL instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SFTL	Left shift	16	No	SFTL S D n1 n2	7
SFTLP		16	Yes		7

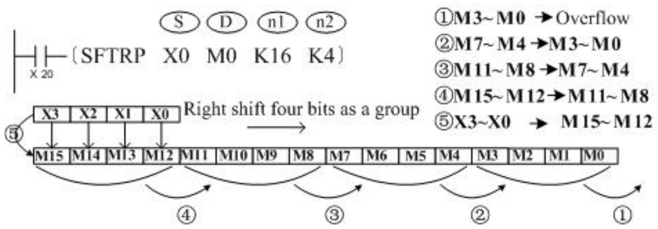
For S bit variables of address started with n2 and D variables of address started with n1, after left shift for n2 bits, the result is saved in D.

The instruction usually uses pulse operation type instruction.

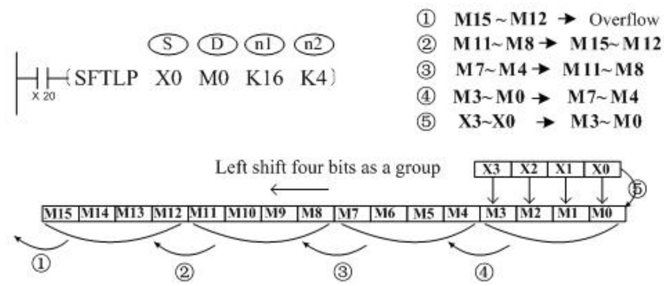
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔	✔	✔	✔											
D		✔	✔	✔											
n1	Constant,n1≤1024														
n2	Constant,n2≤n1														

Programming example:

Example 1 for instruction:



Example 2 for instruction:



SFTR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SFTR	Right shift	16	No	SFTR S D n1 n2	7
SFTRP		16	Yes		7

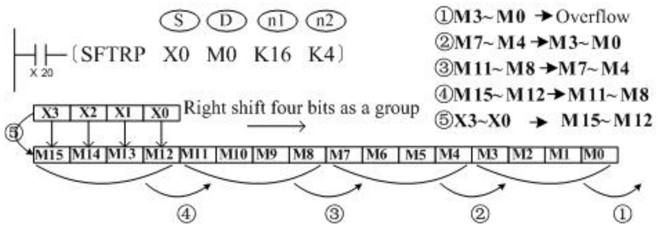
For S bit variables of address started with n2 and D variables of address started with n1, after left shift for n2 bits, the result is saved in D.

The instruction usually uses pulse operation type instruction.

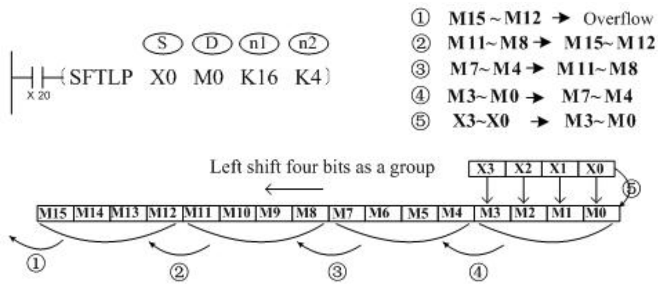
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S		✔	✔	✔											
D	✔	✔	✔	✔											
n1	Constant,n1≤1024														
n2	Constant,n2≤n1														

Programming example:

Example 1 for instruction:



Example 2 for instruction:



SFWR instruction

Instruction Description

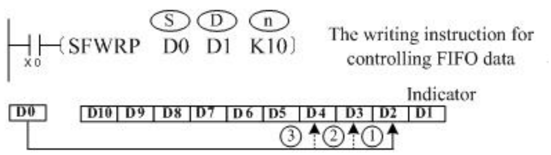
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SFWR	Shift write (the writing instruction for controlling FIFO data)	16	No	SFWR (S) (D)	7
SWFRP		16	Yes		7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(S)					✔	✔	✔	✔	✔	✔	✔	✔	✔		
(D)								✔	✔	✔	✔	✔	✔		
(n)	Constant,2≤n≤2048														

Writing (S) value to the address started with (D). In a (n) FIFO stack, the first numbered device is taken as point. When implementing instruction, the point content is added by 1, and then the device value specified by S will be written to FIFO (D) data tandem location specified by point.

The instruction usually uses pulse operation type instruction.

Programming example:



When X0=1, D0 value is saved to D2, and D1 is set to 1. When X0 is set from OFF to ON again, D0 value is saved to D3, and D1 is set to 2, and so on. If D1 value exceeds n-1, the instruction will not be implemented and carry flag M8022 will be set to 1.

SINH instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
DSINH	SINH calculation for binary floating	32	No	SINH S D	9
DSINHP		32	yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

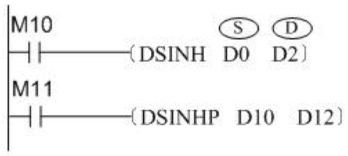
The instruction is to implement SINH calculation for binary floating.The calculation formula is sinh value = $(e^{s}-e^{-s})/2$,where:

S is the binary floating variables for saving SINH value, which is to be calculated;

D is the storage unit for calculation of result

Programming example:

Example 1 for instruction:



When M10 =ON, calculate the SINH value of the binary floating point value in (D1, D0) and save the result to (D3, D2).

SMOV instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SMOV	Bit shift	16	No	SMOV S m1 m2 D n	11
SMOVP		16	Yes		11

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔	✔	✔
m1					✔	✔									
m2					✔	✔									
D								✔	✔	✔	✔	✔	✔	✔	✔
n					✔	✔									

The instruction is driven by contact with five operation variables, where:

S is the data source variable, which is to be copied;

m1 is the starting bit number for transmitting data source with range of 1~4;

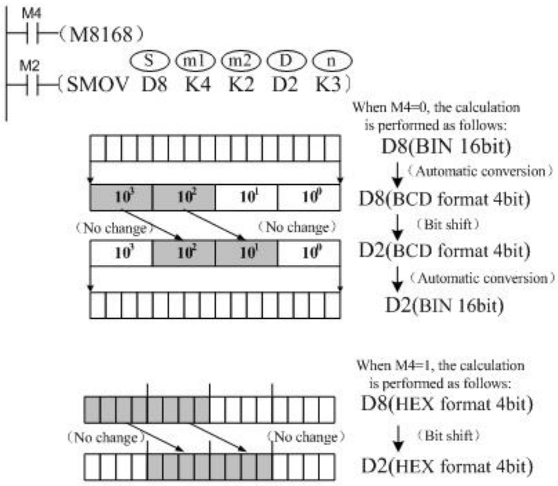
m2 is the bit number for transmitting data source with range of 1~m1;

D is the target variable for transmitting data source;

n is the starting bit of the target variable for transmitting data source with range of m2~4.

The data bit transmission processing is related with the state of special flag M8168. When M8168 is OFF, it is in BCD mode (decimal bit); when M8168 is ON, it is in BIN mode, in which 4-bit is taken as a unit for transmission (hexadecimal bit).

Programming example:



If D8=K1234,D2=K5678, when M8168 is OFF (BCD mode), M2 is set to ON and D2 value is K5128;

When M8168 is ON (BIN mode), D8=H04D 2=K1234,D2=H162E =K5678, M2 is set to ON and D2=H104E=K4174.

SORT instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SORT	Data sorting	16	No	SORT S m1 m2 D n	17

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S													✔		
m1	Constant,1~32														
m2	Constant,1~6														
D													✔		
n					✔	✔							✔		

The instruciton is to implemnet sort operation according to n row parameters for a m1×m2 array, which is described by S m1 m2, and then the result is saved in variable range started with D.

S is the starting unit of the first variable in first line (or called first record);

m1 is the line number of the array, or called record number;

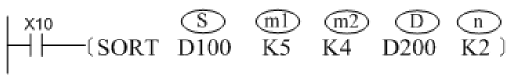
m2 is the row number, or called item number in each record;

D is the starting unit for saving result, occupying following variable unit number is same as that of array before sorting;

n is the array row number, according which the sort operation is implemented.

n Where, m2 value is within the rang of 1~m2.

Programming example:



When X10=ON, sort operation is implemented, and after the implementation, M8029 is set to 1 (program scan period);If it needs re-sorting, X10 should be reset from OFF to ON.

Equivalent table of the above instructions and data examples:

S, m2, m1

Col no. / Row no.	1 Student no.	2 Chinese	3 Mathematics	4 Physics
1	D 100 1	D 105 85	D 110 78	D 115 83
2	D 101 2	D 106 82	D 111 91	D 116 81
3	D 102 3	D 107 77	D 112 89	D 117 88
4	D 103 4	D 108 90	D 113 81	D 118 75
5	D 104 5	D 109 87	D 114 95	D 119 77

Data result after sorting of n =K2 according to the instruction:

D, n =K2

Col no. / Row no.	1 Student no.	2 Chinese	3 Mathematics	4 Physics
1	D200 3	D205 77	D210 89	D215 88
2	D201 2	D206 82	D211 91	D216 81
3	D202 1	D207 85	D212 78	D217 83
4	D203 5	D208 87	D213 95	D218 77
5	D204 4	D209 90	D214 81	D219 75

Data result after sorting of n =K4 according to the instruction:

D, n =K4

列号 / 行号	1 Student no.	2 chinese	3 Mathematics	4 Physics
1	D200 4	D205 90	D210 81	D215 75
2	D201 5	D206 87	D211 95	D216 77
3	D202 2	D207 82	D212 91	D217 81
4	D203 1	D208 85	D213 78	D218 83
5	D204 3	D209 77	D214 89	D219 88

SPD instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SPD	Pulse density	16	No	SPD S1 S2 D	7

The function enable flags M8100~M8105 are respectively used by X000~X005 as enable flag of enhancing function, each of which could be separately set.

1). If function enable flag [M8100~M8105] is OFF, SPD is a basic function:

S1 The input ports of pulse signal should be X0~X5;

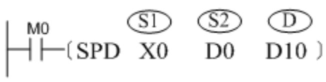
S2 The time of detecting pulse is 1~32767(ms);

D+0: is the pulse number in S2, which is 16bit data.

D+l: The pulse number in this period.

D+2 :Used for detecting remain time (mS).

programming example



X0 is the input port of pulse signal;
D0 is time unit(mS);
D10 is the total pulse number in D0 peorid;
D11 is the counted data in D0 peorid;
D12 is the remain time in D0 period;

2).If function enable flag [M8100~M8105] is ON, SPD is an enhanced function:

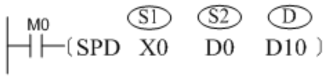
S1 The input ports of pulse signal should be X0~X5;

S2 The time of detecting pulse is 1~32767(ms);

D+0: is the pulse number in S2, which is 16bit data.

D+1, D+2: is the pulse number in each minutes, which is 32bit data.

Programming example:



X0 is the input port of pulse signal;

D0 is unit time 1~32767(mS);

D10 is the total pulse number in D0 period;

D11, D12 is the operation frequency= pulse number in 1 min * 10 (unit is 0.1);

SQR instruction

Instruction Description

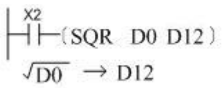
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SQR	BIN radication calculation	16	No	SQR S D	5
SQRP		16	Yes		5
DSQR		32	No		9
DSQRP		32	Yes		9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔							✔		
D													✔		

The instruction is to implement BIN radication operation for S, and the result is saved to D.

S can only be specified as positive. If S is negative, calculation error flag M8067 will be set to ON and the operation will not be implemented, and the calculation result D will be round off.

Programming example:



If D0=K100, D12=K10 when X2 is ON.

If D0=K110, D12=K10 with the decimals rounded off when X2 is ON.

STMR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
STMR	Special timer	16	No	STMR S m D	7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S											✔				
m	Constant,m=1~32767														
S		✔	✔	✔											

The instruction functions to create four special instructions for delay actions according to power flow. Where:

S Timer no.T0~T19 can be used for triggering delay action

m is delay setting in 100 ms ranging from K1 to K32767;

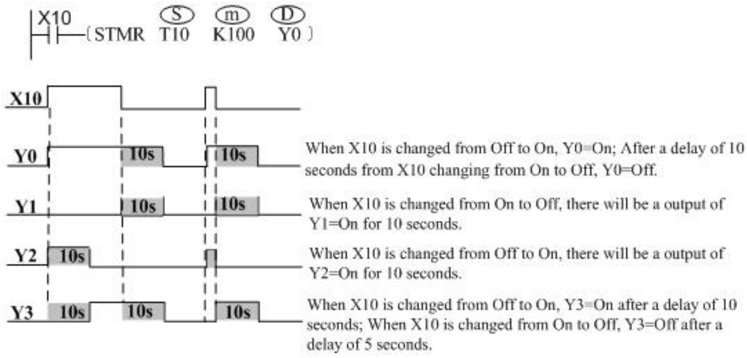
D is starting number for delay action outputting components and occupies 4 consecutive units.

Instruction for use:

The timer used here can't be reused in any of other instructions.

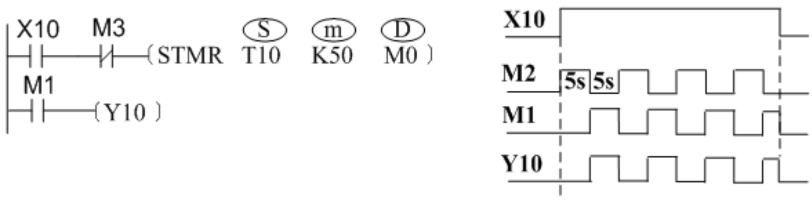
Programming example:

Example 1 for instruction:



Example 2 for instruction:

If component of D is introduced in the instruction energy flow, it is easy to implement oscillator output (the function can also be implemented by using a ALT instruction), which is shown in the following figure:



SUB instruction

Instruction Description

The instruction is to implement subtract operation (A-B=C)for two values.

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SUB	BIN subtration operation	16	No	SUB S1 S2 D	7
SUBP		16	Yes		7
DSUB		32	No		13
DSUBP		32	Yes		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

The instruction is driven by contact with three operation variables. S1 and S2 is subtracted in BIN algebra and saved in D. The involved variables are handled as signed number, whose highest digit is sign bit. 0 is positive number, and 1 is negative.

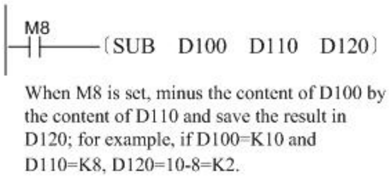
If the calculation result is 0, the 0 flag bit (M8020) will be reset;

When the calculation result exceeds 32,767 (16bit calculation) or -2,147,483,647(32bit calculation), the carry flag bit (M8021) well be reset;

When the calculation result does not exceed -32,768 (16bit calculation) or -2,147,483,648(32bit calculation), the carry flag bit (M8022) well be reset.

When using 32bit calculation, the construction variable address is a low 16bit address, and the adjoining address is a high 16bit address. It should be prevented from repeating or overwriting in the programming.

Programming example:



SUM instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SUM	The instruction is to count the number of instruction with "ON" in soft component	16	No	SUM S D	5
SUMP		16	Yes		5
DSUM		32	No		9
DSUMP		32	Yes		9

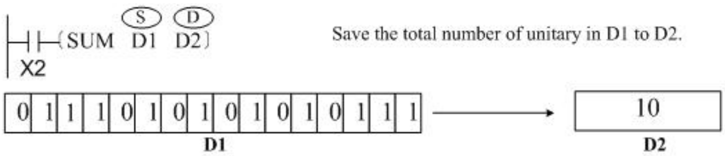
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

The instruction is to count the bit number with "1" in S BIN value, and the result is saved to D.

When using DSUM and DSUMP instructions, the number of bit with "1" in 32bit(S+1,S)is written to D, and D+1 are all set to 0.

If the bits in S are all 0, the zero flag bit M8020 will be set to ON.

Programming example:



SWAP instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
SWAP	Exchange higher and lower byte	16	No	SWAP S	3
SWAPP		16	Yes		3
DSWAP		32	No		5
DSWAPP		32	Yes		5

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔	✔	✔

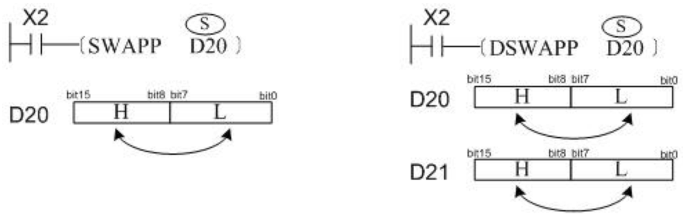
The instruction is to exchange the value between higher and lower byte of specified variableS.

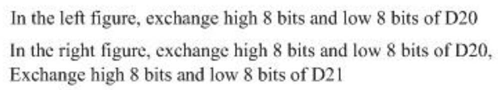
When using 16bit instruction, the exchange operation is implemented between higher 8 bits and lower 8 bits.

When using 16bit instruction, the exchange operation is implemented between higher 8 bits and lower 8 bits.

Notice: the instruction is normally applied as pulse implementation instruction, or if it is applied as continues implementation instruction, the exchange operation will be implemented every time when the program is scanned.

Programming example:





TADD instruction

Instruction Description

Name	**Function**	**Bits(bits)**	**Pulse type**	**Instruction format**	**Step**
TADD	Clock data addition calculation	16	No	TADD S1 S2 D	7
TADDP		16	Yes		7

Operand	**Bit component**				**Word component**										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1											✔	✔	✔		
S2											✔	✔	✔		
D											✔	✔	✔		

The instruction is to implement addition calculation for two sets of clock data in hour/min/second format, and the result is saved in specified variable. Where:

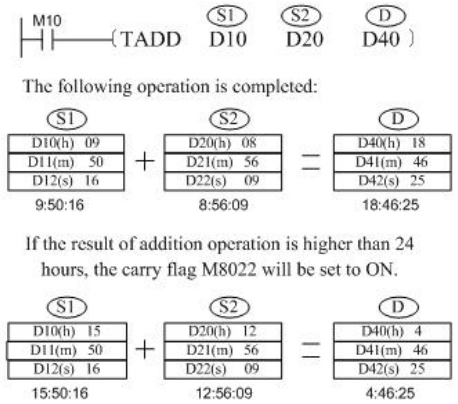
S1 **is time addend occupying three continue variable units, in which hour, minute, and second data are saved in turn.**

S2 **is time addend occupying three continue variable units, in which hour, minute, and second data are saved in turn.**

D **is time addition occupying three continue variable units, in which hour, minute, and second data are saved in turn.**

If the calculation result exceeds 24 hours, the carry flag M8022 is set to 1 and the actual displayed time will be subtracted with 24:00:00; If the calculation result is 00:00:00, zero flag M8020 is set to 1;

Programming example:



TANH instruction

Instruction Description

Name	**Function**	**Bits(bits)**	**Pulse type**	**Instruction format**	**Step**
DTANH	TANH calculation for binary floating	32	No	TANH S D	9
DTANHP		32	Yes		9

Operand	Bit component				Word component											
	X	Y	M	S	K	H	E	KnX	KnY	KnM	KnS	T	C	D	V	Z
S					✔	✔	✔							✔		
D														✔		

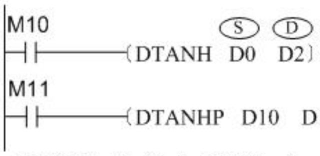
The instruction is to implement TANH calculation for binary floating.The calculation formula is tanh value $(e^{s}-e^{-s})/(e^{s}+e^{-s})$,where:

S **is the binary floating variables for saving TANH value, which is to be calculated;**

D **is the storage unit of the calculation result**

Programming example:

Example for instruction:



When M10=ON, calculate the TANH value of binary floating point value in (D1, D0) and save the result to (D3, D2).

TCMP instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TCMP	Absolute Positioning	16	No	TCMP S1 S2 S3 S D	9
TCMPP		16	Yes		9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S3					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S											✔	✔	✔		
D		✔	✔	✔											

The instruction is to implement comparison between specified hour/min/second value and internal real-time clock, and output the comparison result.Where:

S1 **is the "hour" in specified comparison time with the range of 0~23;**

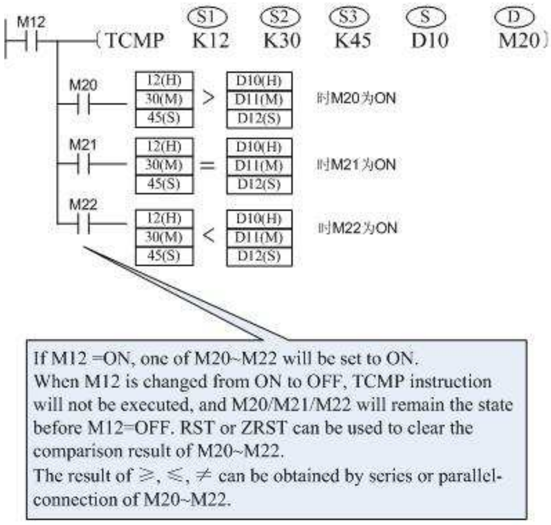
S2 **is the "minute" in specified comparison time with the range of 0~59;**

S3 **is the "second" in specified comparison time with the range of 0~59;**

S **is the starting address of the real-time clock time register and normally the saving unit after clock read TRD or MOV instruction.**

D **is the starting address of storage variable for comparing result, occupying following three variable units;**

Programming example:



TKY instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TKY	Cross key input	16	No	TKY S D1 D2	7
DTKY		32	No		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S	✔	✔	✔	✔											
D1								✔	✔	✔	✔	✔	✔	✔	✔
D2		✔	✔	✔											

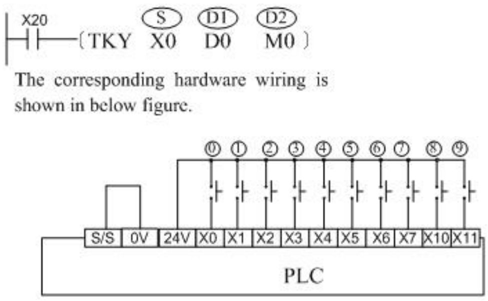
The instruction is to specify ten continuous variable units (such as X input port), which represents 0~9 key in decimal. When pressing key event occurs (the state is ON), the 4-digit decimal value (0~9999) can be inputted according to the order of pressing operation. If using 32bit instruction, the 8-digit decimal value (0~99999999) can be inputted.

S **Is the starting input port of pressing key, occupying the following ten bit units (such as X port);**

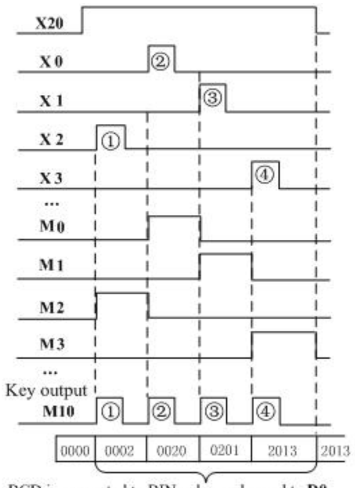
D1 **Is the storage unit for inputted value;**

D2 **Is the temp starting unit for state of current pressing key group, occupying the following eleven bit units**

Programming example:



If you want to input "2013", just pressing key 2, 0, 1, 3 in order. The operation of PLC internal variable is shown as following figure.



BCD is converted to BIN value and saved to **D0**.

Set by parameters in an instruction, X0~X11 respectively correspond to numeric keys 0~9; M0~M9 correspond to the status of keys; Key output unit will be reset whenever a key is pressed.

Key values (e.g. 2013) are converted to BIN and saved to the designated D1 unit D; (D0=0x7DD), D0 will never change even when the power flow turns OFF.

When several keys are pressed simultaneously, the key which is firstly detected is valid; if the number entered is more than 4 digits, the first entered number will overflow and only a 4-digit number is left.

If using 32bit instruction (DTKY), and D1 occpies 32bit variable. For the above case, they are D1, D0, whih is higher word and lower word respectively.

TO instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TO	Write BFM	16	No	TO m1 m2 D n	9
TOP		16	Yes		9
DTO		32	No		17
DTOP		32	Yes		17

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D								✔	✔	✔	✔	✔	✔	✔	✔
m1=0~7; m2=0~32767; n=1~32767; When D component is specified, K1~K4 are availabe (16bit); K1~K8 are available (32bit); m1,m2,n do not support character D register.															

The instruction is used to implement data writing operation to BFM register in specially extended module. Where:

m1 is the address serial number of the special extended module, whose value ranges 0~7. 0 is the closest to the main module and the number goes on. Maximum of 8 special modules are allowed.

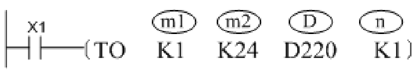
m2 is the register address code of BFM inside the special module. It has values ranging from 0~32767;

D Is the parameter register address in main module, and the parameters can taken as the source of writing operation data. When the register for writing operation is more than one, the following units can be occupied;

D The number of written parameters with the range of 1~32767, which could be written in turn according to register address.

Programming example:

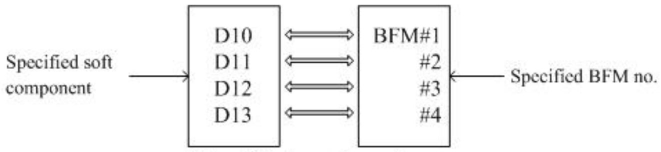
Example 1 for instruction:



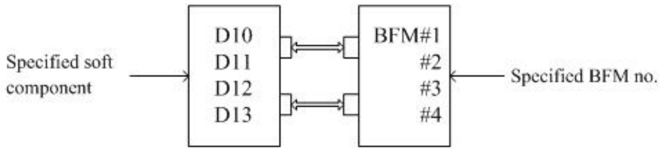
When X1 is ON, the data saved in PLC D220 register can be written to the No. 24 address in #1 special module, each for one time. When X1 is OFF, the operation will not be implemented.

When using 32bit instruction, the specified address is lower 16bit address, +1 is higher 16bit address. For example:

n Is the number of operating data, n=2 meaning 2 Word (16bit instruction); n=1 meaning 2 Word (32bit instruction). So please pay attention to that n=2 (16bit instruction) and n=1(32bit instruction) have the same meanings. For example:



When 16bit instruction n=4



When 32bit instruction n=2

The description of FROM/TO instruction:

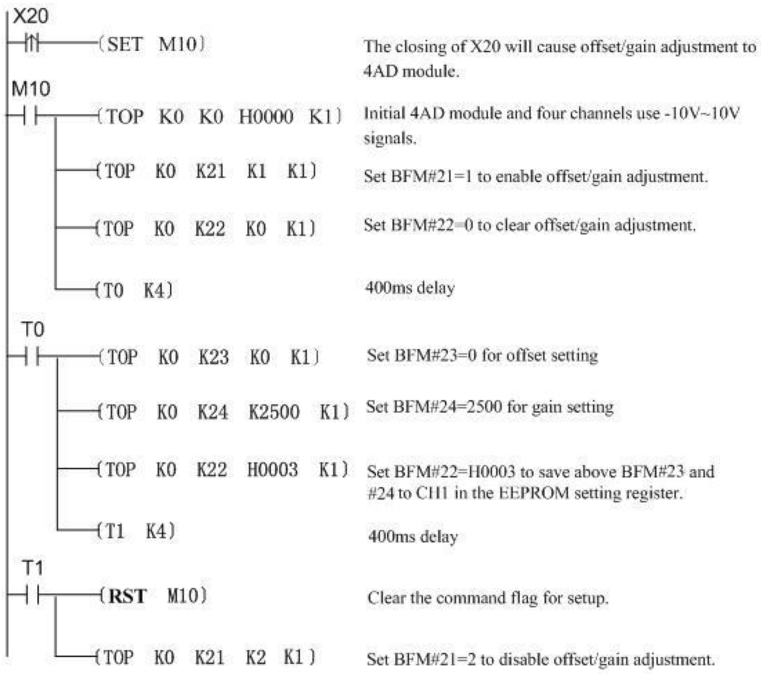
1. M8164 (the changeable transmission number mode of FROM/TO instruction), When M8164=ON and implementing FROM/TO instruction, the content fo special data register D8164 (the specified transmission number register of FROM/TO instruction) will be taken as transmission number n for processing;

2. Accessing extended module with FROM/TO instruction is a time-consuming operation. When multiple FROM/TO instruments is implemented or multiple buffer memory data is transmitted, the PLC scanning period will be extended. In order to prevent overtime, you can add WDT instruction for extending monitor timer cycle before FROM/TO instruction, or stagger the operation time of FROM/TO instruction, or use pulse operational instruction.

3. For the use of connection method and input/output numbering of special module, please refer to the manual attached with them.

Example 2 for instruction:

An example code of setting offset/gain operation for H2U-4AD module CH1 channel with #0 number address is listed as following:



TRD instruction

Instruction Description

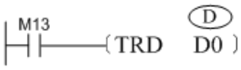
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TRD	Read clock data	16	No	TRD D	3
TRDP		16	Yes		3

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D											✔	✔	✔		

The instruction is to read the PLC built-in real-time clock, including year, month, day, hour, minute, second, and week, which will be saved in specified register.

where, D is the starting storage unit for saving read time, occupying seven continous variable units. The time data order is year, month, day, hour, minute, second, and week, which should be saved with increament order.

Programming example:



The operation is shown as following:

Item	System variables		After operation D
Year (0~99)	D8018	→	D0
Month (1~12)	D8017	→	D1
day (1~31)	D8016	→	D2
Hour (0~23)	D8015	→	D3
Minute (0~59)	D8014	→	D4
Second (0~59)	D8013	→	D5
Week (0 ~ 6)	D8012	→	D6

Note: normally, it is recommended to use PLC clock. Using the data in D register, which is read from clock with TDR instruction, instead of using D8012~D8018 value directly.

TSUB instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TSUB	Clock data addition calculation	16	No	TSUB S1 S2 D	7
TSUBP		16	Yes		7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1											✔	✔	✔		
S2											✔	✔	✔		
D											✔	✔	✔		

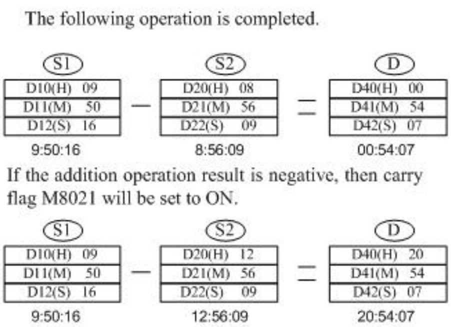
S1 The instruction is to implement subtraction calculation for two sets of clock data in hour/min/second format, and the result is saved in specified variable. Where:

S2 is time minuend occupying three continue variable units, in which hour, minute, and second data are saved in turn.

D is time subtract result occupying three continue variable units, in which hour, minute, and second data are saved in turn.

If the calculation result is negative, the borrow flag M8021 is set to 1 and the actual displayed time will be added with 24:00:00; If the calculation result is 00:00:00, zero flag M8020 is set to 1;

Programming example:



TTMR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TTMR	Teach timer	16	No	TTMR D n	5

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D													✔		
n					✔	✔									

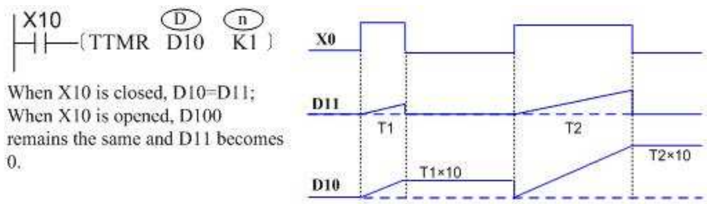
The instruction is to implement multiplication calculation for holding time of pressing specified input key and n , and then the result is saved to variable D, which is usually used to set parameters. Where:

D is the multiplication result of holding time of pressing key in second and n, and D content is not changed after releasing key;
D£«1 unit is used for saving holding time of pressing key in 100ms, and it will be reset to 0 after releasing key;

n For setting multiple, please pay attention that the actual multiple is 10n, where n=0~2.
When n=K0, the actual multiple is 1;
When n=K1, the actual multiple is 10;
When n=K2, the actual multiple is 100;

Programming example:

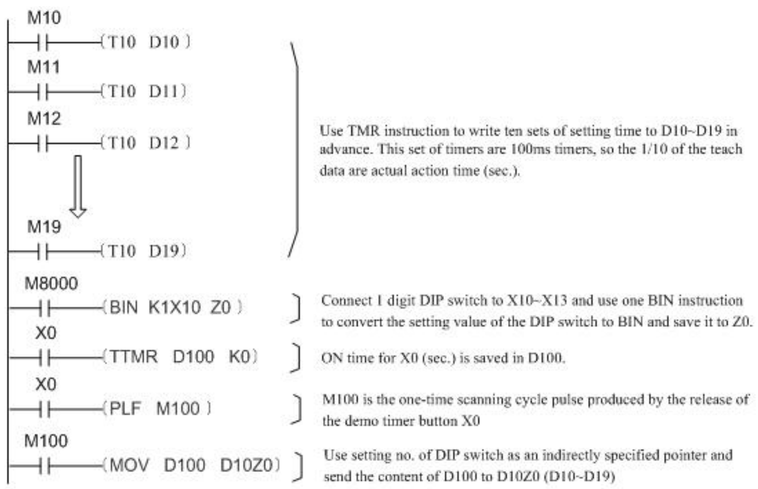
Example 1 for instruction:



If holding time of pressing key X10 is T seconds, the relationships between D10, D11, and n are listed as following:

n	D10	D11(unit: 10 ms)
K0(unit: sec)	1×T	D11=D10×10
K1(unit: 100 ms)	10×T	D11=D10
K2(unit: 10 ms)	100×T	D11=D10/10

Example 2 for instruction:



TWR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TWR	Write clock data	16	No	TWR S	3
TWRP		16	Yes		3

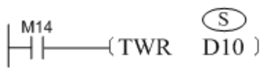
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S											✔	✔	✔		

The instruction is to write the seven data of the specified clock data S (including year / month / day / hours / minutes / seconds / week) into PLC built-in data of real-time clock. Where:

S is initial memory cell for saving reading time, and occupies seven continuous variable units. Addresses storage data in growing number: year, month, day, hour, minute, second, week, and so on.

Programming example:

Example 1 for instruction:



The operation is shown as following:

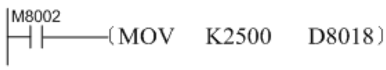
Item	Data source D		System variables
Year (0~99)	D0	→	D8018
Month (1~12)	D1	→	D8017
Day (1~31)	D2	→	D8016
Hour (0~23)	D3	→	D8015
Minute (0~59)	D4	→	D8014
Second (0~59)	D5	→	D8013
Week [0(day)~6]	D6	→	D8012

Note that the seven data are whole written when clock is written. Any variable can not be lacked when you preset the value. If week is not written, the default is 0 for Sunday; if month is not written, the month variable is 0, and PLC believes that the month you provide is wrong. Thus the clock change is invalid.

Once M8017 produce one ON, PLC internal clock does ± 30 correction action. Where the correction means that when the PLC's internal clock second hand is in 1~29, clock will be automatically classified as "0" seconds and minute hand does not act; in 30 ~9, it will also be automatically classified as "0" seconds, minutes plus 1 minute.

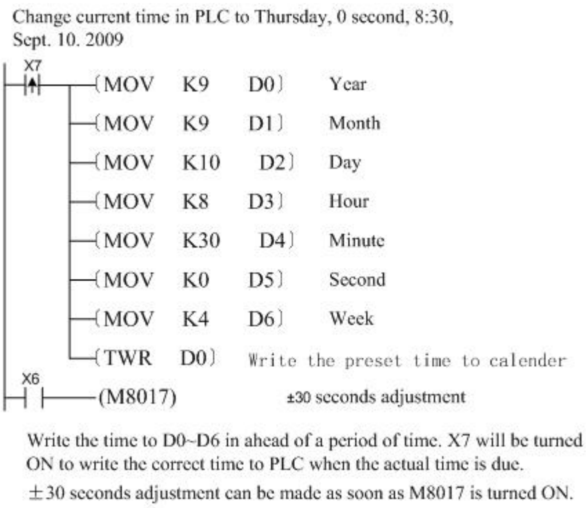
M8015 set ON to stop the clock timing.

In the usual case it shows only 2 digits (for example: in 2009 only show 09), If you hope that "year" shows four digits format, execute the following statements in one scan cycle:



If D8018=09 at the first time, D8018=2009 after switch. PLC internal clock is as follows.

Example 2 for instruction:



Note: Usually you have to modify PLC clock. Write the clock into D8013~D8019 by TWR instruction. Do not use the MOV instruction for direct assignment of the D8012~D8018.

TZCP instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
TCZP	Clock data area comparison	16	No	TZCP S1 S2 S D	9
TZCPP		16	Yes		9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1											✔	✔	✔		
S2											✔	✔	✔		
S											✔	✔	✔		
S		✔	✔	✔											

The instruction is making a comparison of built-in real-time clock data and specified two group hour / minute / second preset value range, and exports the comparison result Where:

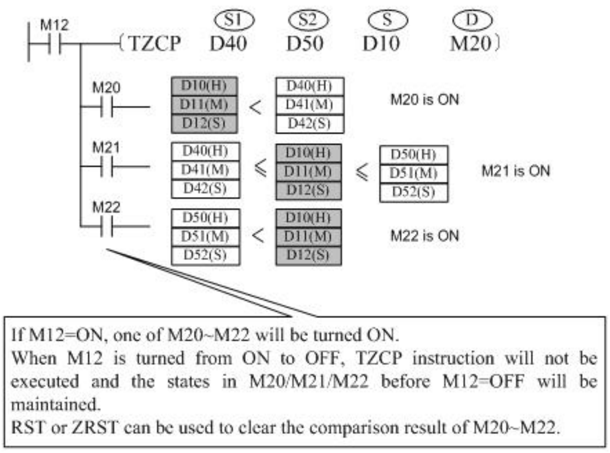
S1 is the specified lower time limit. It occupies three continues variable units and that is hour, minute, second data.

S2 is the specified upper time limit. It occupies three continues variable units and that is hour, minute, second data.

S is the starting address of the real-time clock time register and normally the saving unit after clock read TRD or MOV instruction.

D is the starting address of storage variable for comparing result, occupying following three variable units;

Programming example:



WAND instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
WAND	Logic AND	16	No	WAND (S1) (S2) (D)	7
WANDP		16	Yes		7
DWAND		32	No		13
DWANDP		32	Yes		13

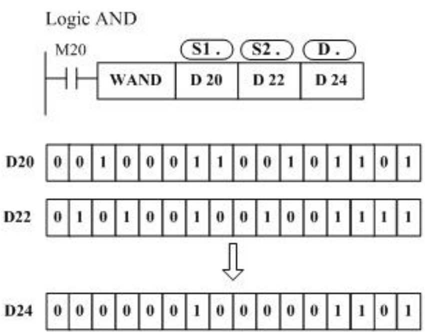
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
(S1)					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
(S2)					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
(D)								✔	✔	✔	✔	✔	✔	✔	✔

When the instruction runs, every BIN value digit of (S1) and (S2) use "logic and" operation. The result is stored in variable.

The ruler of "logic and" is that result is zero when anyone is zero.

1∧1=1 1∧0=0 0∧1=0 0∧0=0

Programming example:

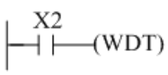


WDT instruction

Instruction Description

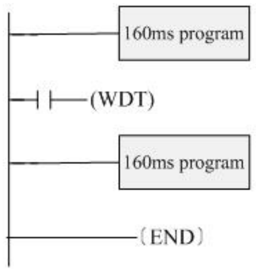
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
WDT	monitoring timer's refresh	16	No	None(needn't driver connect point's single instruction)	1
WDTP		16	Yes		1

The PLC system have a timer ,which are used to monitor whether the user's program execution time is a time-out. If time is out , the user program will stop and report alarm .Executing WDT instruction can reset monitoring timer, and makes the monitoring timer restart timing ,avoid the time-out error.



If the operation of user's program is too complex (for example, too many Cycle of calculation), an error may occur when the implementation of programming running out . If necessary, the program can use WDT instruction (for example, between the FOR ~ NEXT instruction can insert the instructions); If the program's scaning time is longer than the value of D8000 (default 200ms), we can insert program between the WDT instructions. The program will be divided into pieces ,every piece¡¯s scaningtime is less than 200ms or change the setting value of D8000.

Programming example:



This program scanning time is 320ms. we can divide program into two parts with the WDT instruction, so that each part of the program scanning time is bellow 200ms.

WOR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
WOR	Logic OR	16	No	WOR S1 S2 D	7
WORP		16	Yes		7
DOR		32	No		13
DORP		32	Yes		13

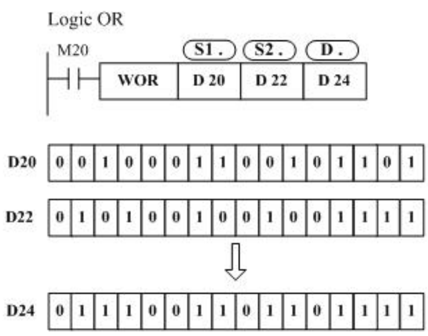
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

When the instruction runs, recognize every figure of S1 and S2 BIN value as "logic or" operation. The result is stored in D variable.

The rule of logic "or" is that the result is one when anyone is one.

1V1=1 1V0=1 0V1=1 0V0=0

Programming example:



WR3A instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
WR3A	Write analog module	16	No	WR3A m1 m2 D	7
WR3AP		16	Yes		7

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
m1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
m2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

This instruction provides writing instructions of Mitsubishi Type FX0N~3A analog output module. Where:

m1 **is the number of special module; (K0~K7).**

m2 **is analog input channel; (K1)**

D **Is the parameter register address in main module, and the parameters can taken as the source of writing operation data.**

Programming example:

TO instruction can also be used on writing of FX0N ~3A analog output value.

WSFL instruction

Instruction Description

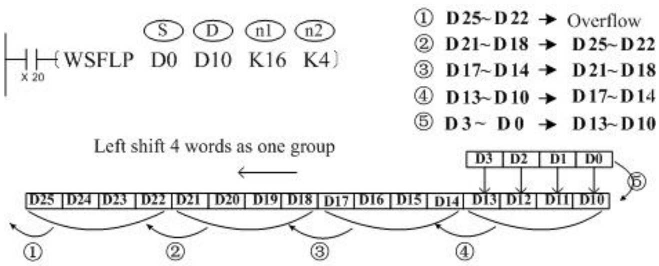
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
WSFL	Shift right by word	16	No	WSFL S D n1 n2	9
WSFLP		16	Yes		9

Shift n2 **variables of** S **initiation address and** n1 **variables of** D **initiation address to left** n2 **words in word units..**

The instruction usually uses pulse operation type instruction.

Operand	Bit component				Word component									
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔		
D								✔	✔	✔	✔	✔	✔		
n1	Constant,n1≤2048														
n2	Constant,n2≤n1														

Programming example:



WSFR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
WSFR	Shift right by word	16	No	WSFR S D n1 n2	9
WSFRP		16	Yes		9

For S **bit variables of address started with** n2 **and** D **variables of address started with** n1 **, after left shift for** n2**bits, the result is saved in.**

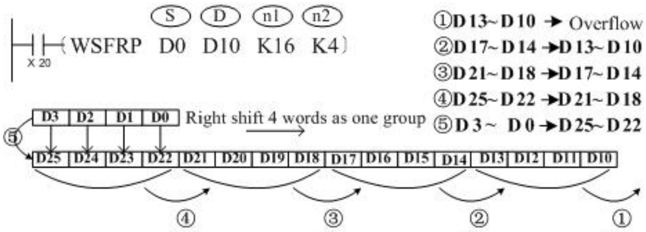
The instruction usually uses pulse operation type instruction.

Operand	Bit component				Word component									
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S							✔	✔	✔	✔	✔	✔	✔		
D								✔	✔	✔	✔	✔	✔		
n1	Constant,n1≤2048														
	Constant,n2≤n1														

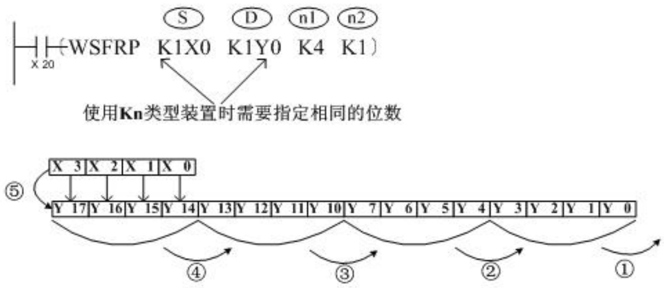
n2	

Programming example:

Example 1 for instruction:



Example 1 for instruction:



Right shift operation in one-time scanning is performed according to the following no. 1~5.

1: Y3~Y0 → carry bit

2: Y17~Y14 → Y13~Y10

3: Y13~Y10 → Y7~Y4

4: Y7~Y4 → Y3~Y0

5: X3~X00 → Y17~Y14 Completed

WXOR instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
WXOR	Logic XOR	16	No	WXOR S1 S2 D	7
WXORP		16	Yes		7
DXOR		32	No		13
DXORP		32	Yes		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D								✔	✔	✔	✔	✔	✔	✔	✔

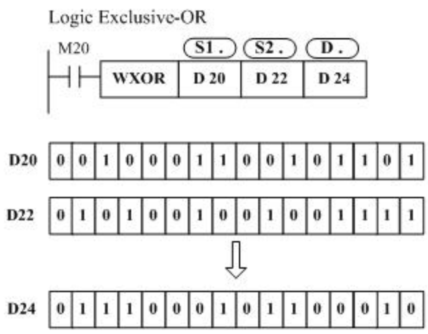
When the instruction runs, it will take "logic of exclusive or" operation corresponding BIN value of S1 and S2. The result is stored in the D variable.

The rule of logic 'exclusive OR' (XOR) operation is 0 when the both results are same or 1 when the both results are different.

1^1=0 1^0=1 0^1=1 0^0=0

The three instruction operands refer to the variable type as the following table. W hen the instruction is 32bit, the register variables will occupy the following two units:

Programming example:



XCH instruction

Instruction Description

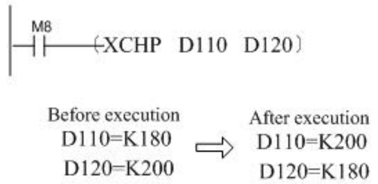
Name	Function	Bits(bits)	Pulse type	Instruction format	Step
XCH	Date exchange	16	No	XCH S D	5
XCHP		16	Yes		5
DXCH		32	No		9
DXHCP		32	Yes		9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S								✔	✔	✔	✔	✔	✔		
D								✔	✔	✔	✔	✔	✔		

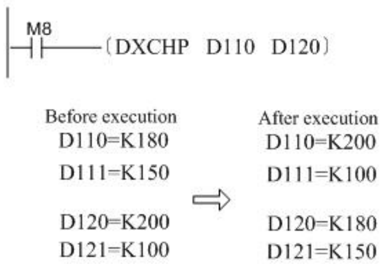
Need contact drive, there are tow operating variables. Exchange the number of S and D.

Programming example:

Example 1 for instruction:



Example 2 for instruction:



When special variable M8160£½1,and the dress of D and S are same, the resule of opreation will be the exchange of high 8 bits and low 8 bits, 32 bits instruction is same. Equval to the SWAP instruction opreation. General realize by the swap instruction

ZCP instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ZCP		16	No		7

ZCPP	Regional comparison	16	Yes	ZCP S1 S2 S D	7
DZCP		32	No		13
DZCPP		32	Yes		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
D		✔	✔	✔											

Need contact drive, there are four operating variables. When the control of power flow is effective , conduct algebraic comparison by the signed number, S1 S2 is interval, take S 's position in the interval to be the result, take the result into three contiguous variables which take D as the starting address. Where:

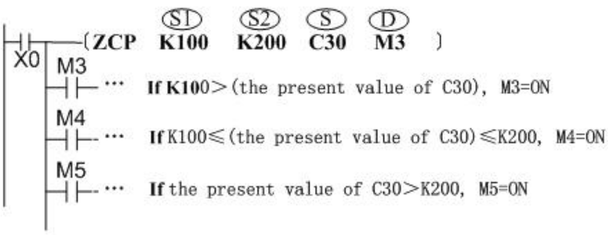
S1 is lower limit of comparison area

S2 is upper limit of comparison area

S is comparison variable

D is the storage cell of comparison result, it will occupy three continuous bit variables

Programming example:



If X0=ON, one of M3~M5 will become ON.
If X0 is turned from ON to OFF, ZCP instruction will not be executed and M3~M5 remain the state before X0=OFF. RST or ZRST can be used to clear the comparison result of M3~M5.

ZRN instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ZRN	Regression through the Origin	16	No	ZRN S1 S2 D	9
DZRN		32	No		13

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S3	✔	✔	✔	✔											
D		✔													

When servo driver cooperates with PLC, this instruction is used to enable actuator to move toward DOG with designated impulse speed and impulse-output port, until condition of encountering original point is satisfied .

S1 is start speed of the regression through the origin action which range is 10~32,767Hz when in 16bit modle,while 10~100,000Hz in 32bit modle.

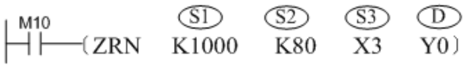
S2 is crawling speed when original point signal turns ON which is ranging 10Hz to 32767Hz.

S3 is input of DOG,Although signal XYMS is well,timeliness of signal X function best.

D is start address of impulse output.With regard to 3624MT/2416MT in the type of MT,only Y0 and Y1 can be allocated,while others can only be allocated Y0/Y1/Y2.As to type of MTQ,Y0/Y1/Y2/Y3/Y4 can be all allocated.

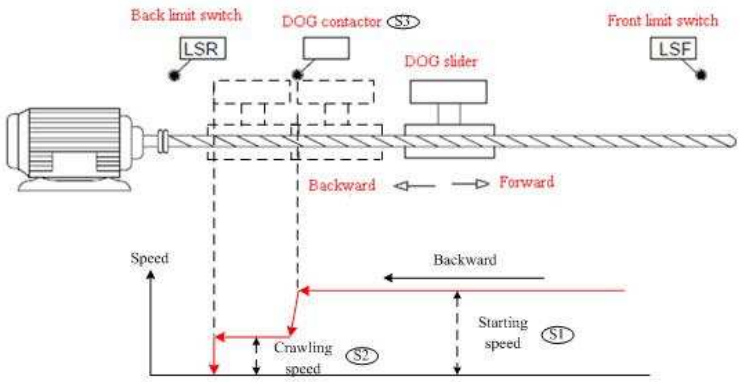
When DRVI-FNC158 and DRVA-FNCl59 are excuted,contrller can calculate pulse number of positive rotation and inversions and save them to register [D8141, D8140] (Y000) and [D8143, D8142] (Y001). But data in that register will disappear after power failure, so instruction ZRN must be executed when system is power on and initialized run, so the data of original position of mechanical movement can be read in beforehand.

Programming example:



This instruction means that, after M10 turns ON,PLC send out pulses at speed of 100Hz from out-put port Y0 making stepper motor draw back toward original point. While when DOG turns ON(when slide block just touch contactor) output frequency turns to 80Hz creeping at lower speed, until DOG turns OFF again, and at the same time Y0 stops outputing pulse, input 0 to current register Y000:[D8141,D8140],Y001:[D8143 , D8142]. In addition,when M8140 turns ON,Y0 resets. Whereafter, M8029 is set ON,at the same Y000:[M8147],Y00l:[M8148] turns OFF.

See figure below:



During this command is excuted, systemic variables concerned are:
1. D8141(high-order), D8140(low-order):Y000 outputs value of current register(using 32 bit)
2. D8143(high-order), D8142(low-order):Y001 outputs value of current register(using 32 bit)
3. M8145 :
4. M8146 :
5. M8147 :
6. M8148 :

Since servo driver has the function of power-fail-safeguard towards location information, this command does not need to excute after power-on every time. Meanwhile, for servo driver can only move one way, movement of backing to original point must be done before DOG..

Notice:
Positioning instruction (ZRN/PLSV/DRVI/DRVA) can be reused in the program, but do not output to the same port;
If the drive power flow for an instruction turns OFF and ON again, it can only be driven after one operation cycle when status bit (Y 000： [M8147], Y001: [M8148]) turns OFF.
When positioning instruction is driven again, there should be at least one cycle of OFF time. If the re-drive is implemented in the time less than above condition, there will be calculation error when firstly implementing calculation instruction.

ZRST instruction

Instruction Description

Name	Function	Bits(bits)	Pulse type	Instruction format	Step
ZRST	Reset all	16	No	ZRST D1 D2	5
ZRSTP		16	Yes		5

Operand	Bit component				Word component									
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
D1		✔	✔	✔							✔	✔	✔		
D2		✔	✔	✔							✔	✔	✔		

Clear all variable between D1 and D2,which can be word-variable also can be bit-variable like Y\M\S.

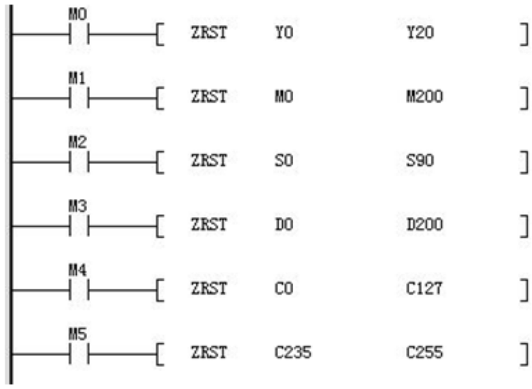
request:

D1 and D2 must be the same kind of soft component.

As to serial number,D1 cannot bigger than D2,if they are the same,just reset the prescribed soft component.

This instruction is 16bit,but D1 and D2 can be allocated 32 bit counters and D1 and D2 should be 16bit or 32 bit at the same time.

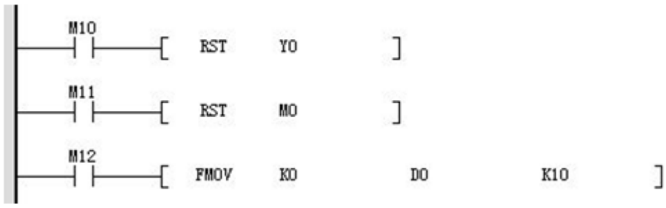
Programming example:



Additional remarks:

Device of bitY¡¢M¡¢S and device of word T,C,D can also be reseted by RST separately.

Device of T\C\D and device of T\C\D,including bit register KnY\KnM\KnS,can also be cleared from multi-points by FMOV Where:



AND series of contact compare instructions

Instruction Description

Name	Function	Bit(bits)	Pulse type	Instruction format	Step
AND=	(s1)=(s2) TRUE	16	No		5
ANDD=	(s1)=(s2) TRUE	32	No		9
AND>	(s1)>(s2) TRUE	16	No		5
ANDD>	(s1)>(s2) TRUE	32	No		9
AND<	(s1)<(s2) TRUE	16	No		5
ANDD<	(s1)<(s2) TRUE	32	No	AND☼ S1 S2	9
AND<>	(s1)<>(s2) TRUE	16	No	☼Comparison operators such as in =,>,=,<=,<> etc.	5
ANDD<>	(s1)<>(s2) TRUE	32	No		9
AND<=	(s1)<=(s2) TRUE	16	No		5
ANDD<=	(s1)<=(s2) TRUE	32	No		9
AND>=	(s1)>=(s2) TRUE	16	No		5
ANDD>=	(s1)>=(s2) TRUE	32	No		9

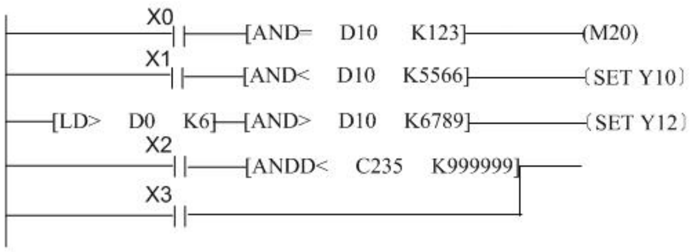
Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

Some other logic operations have been made before the instruction. The instruction compares two operands and makes the result participate in the operation of program energy flow in the form of logic state. All the variables used in the comparison can be regarded as signed number, among them:

S1 **is the data source or data variable unit 1 for comparison**

S2 **is the data source or data variable unit 2 for comparison**

The programming example



If X0=ON and the value in D10=K123, M20=ON.

If X0=ON and the value in D10 < K5566, Y10=ON and holds.

If the value in D0 > K6 and K6789, Y12=ON and holds.

If X2=ON and the value in C235 < K9999999 or X3=ON, Y15=ON and holds.

-

If the operands are two 32bits-width counters, you should use the instruction ANDD which is designed for 32bits-width-operands, otherwise an error would happen. When 32-bit counter(C200~C255) compares this instruction, be sure to use 32-bit instructions

LD series of contact compare instructions

Instruction Description

Name	Function	Bit(bits)	Pulse type	Instruction format	Step
LD=	(s1)=(s2) TRUE	16	No	LD☼ S1 S2 ☼Comparison operators such as in=,>,=,<=,<> etc.	5
LDD=	(s1)=(s2) TRUE	32	No		9
LD>	(s1)>(s2) TRUE	16	No		5
LDD>	(s1)>(s2) TRUE	32	No		9
LD<	(s1)<(s2) TRUE	16	No		5
LDD<	(s1)<(s2) TRUE	32	No		9
LD<>	(s1)<>(s2) TRUE	16	No		5
LDD<>	(s1)<>(s2) TRUE	32	No		9
LD<=	(s1)<=(s2) TRUE	16	No		5
LDD<=	(s1)<=(s2) TRUE	32	No		9
LD>=	(s1)>=(s2) TRUE	16	No		5
LDD>=	(s1)>=(s2) TRUE	32	No		9

Operand	Bit component				Word component										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z

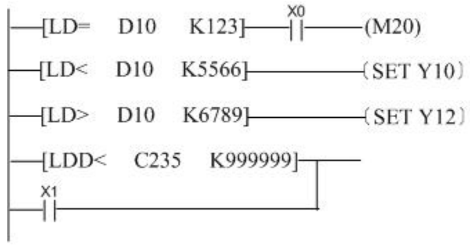
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

The instruction LD would compare one operand to another, and output a logical state as comparing result. Observe that both of the operands are singed, among them:

S1 **is the data source or data variable unit 1 for comparison**

S2 **is the data source or data variable unit 2 for comparison**

The programming example



If the content of D10= K123 and X0=ON, M20=ON.

If the content of D10 < K5566, Y10=ON and holds.

If the content of D10 > K6789, Y12=ON and holds.

If the content of C235 < K999999 or X1=ON, Y15=ON.

-

If the operands are two 32bits-width counters, you should use the instruction LDD which is designed for 32bits-width-operands, otherwise an error would happen. When 32-bit counter(C200~C255) compares this instruction, be sure to use 32-bit instructions

OR series of contact compare instructions

Instruction Description

Name	**Function**	**Bit(bits)**	**Pulse type**	**Instruction format**	**Step**
OR=	(s1)=(s2) TRUE	16	No		5
ORD=	(s1)=(s2) TRUE	32	No		9
OR>	(s1)>(s2) TRUE	16	No		5
ORD>	(s1)>(s2) TRUE	32	No		9
OR<	(s1)<(s2) TRUE	16	No		5
ORD<	(s1)<(s2) TRUE	32	No	OR☼ S1 S2	9
OR<>	(s1)<>(s2) TRUE	16	No	☼ Comparison operators such as in=,>,=,<=,<> etc.	5
ORD<>	(s1)<>(s2) TRUE	32	No		9
OR<=	(s1)<=(s2) TRUE	16	No		5
ORD<=	(s1)<=(s2) TRUE	32	No		9
OR>=	(s1)>=(s2) TRUE	16	No		5
ORD>=	(s1)>=(s2) TRUE	32	No		9

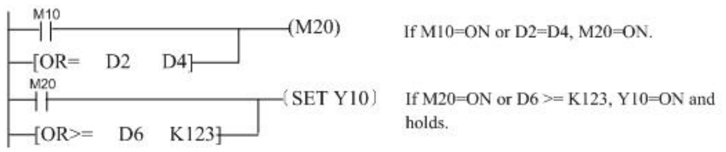
Operand	**Bit component**				**Word component**										
	X	Y	M	S	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
S1					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
S2					✔	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔

The instruction compares two operands and makes the result participate in the OR operation of program energy flow in the form of logic state. All the variables used in the comparison can be regarded as signed number, among them:

S1 **is the data source or data variable unit 1 for comparison**

S2 **is the data source or data variable unit 2 for comparison**

The programming example



•

For calculator which is used by numbers with a bandwidth of 32bit, ORD instruction should also be 32bit, otherwise an error would happen. When 32-bit counter(C200~C255) compares this instruction, be sure to use 32-bit instructions

A series of H2U/H1U PLC's communication hardware config

COM0's communication protocol config

COM0 protocol	setting of D8116	semiduplex/full-duplex mode	COM0communication format
Download/HMI monitor protocol	01h	JP0 OFF :download protocol JP0 ON:when state from STOP to ON ,decided byD8116	fixup
MODBUS-RTU slave station	02h	semiduplex	decided by D8110
MODBUS-ASC slave station	03h	semiduplex	decided by D8110
Other protocol（contains RS instruction）	nonsupport		

COM1's communication protocol config

COM1 protocol	setting of D8126	semiduplex/full-duplex mode	COM1communication format
HMI monitor protocol	01h	semiduplex	fixable setting by PLC system software
1:1 parallel connection	50h	semiduplex	

protocol host station			
1:1 parallel connection protocol slave station	05h	semiduplex	
N:N protocol host station	40h	semiduplex	
N:N protocol slave station	04h	semiduplex	
Computer link protocol	06h	semiduplex	setting by D8120**
MODBUS-RTU slave station	02h	semiduplex	
MODBUS-ASC slave station	03h	semiduplex	
MODBUS RTU instruction	20h	semiduplex	
MODBUS-ASC instruction	30h	semiduplex	
RS instruction	10h or 00h	setting by Bit10 of D8120 1:semiduplex,standard config port is RS485 0:full-duplex mode,interface of RS 232C /RS422-BD extended card	

Definement of D8110 and D8120 which is communication format config

protocol name	baud rate	data length	parity	stop bit
N:N protocol	limit 38400	limit 7	limit even parity	limit 1bit
1:1 parallel connection protocol	limit 19200	limit 7	limit even parity	limit 1bit
HMI monitor	limit 9600	limit 7	limit even parity	limit 1bit

protocol				
Computer link protocol	Setting by Bit7 to Bit4 of D8110 in COM0.	Setting by Bit0 of D8110 in COM0 :	Setting by Bit2 to Bit1 of D8110 in COM0.	Setting by Bit3 of D8110 in COM0.
MODBUS-RTU slave station	Setting by Bit7 to Bit4 of D8120 in COM1: 0011b-300BPS 0100b-600BPS 0101b-1200BPS 0110b-2400BPS 0111b-4800BPS 1000b-9600BPS 1001b-19200BPS 1010b-38400BPS 1011b-57600BPS 1100b-115200BPS	0b-7Bits 1b-8Bits Note: MODBUS-RTU slave station protocol and instruction only support 8-bit data length，otherwise communications error be caused of	Setting by Bit2 to Bit1 of D8120 in COM1: 00b-none parity (N) 01b-odd parity (O) 11b-even parity (E)	Setting by Bit3 of D8120 in COM1 : 0-1Bits 1-2Bits
MODBUS-ASC slave station				
RS instruction				
MODBUS RTU instruction				
MODBUS-ASC instruction				

Communication protocol that series of H2U/H1U PLC's system software to confirm principle of COM1 port:

1.When state from STOP to RUN,system will search setting about config word D8126 in user program and ensured the communication protocol.

2. After PLC run,communication protocol changeless ,though PLC program modified protocol config word D8126.

PLC's download/HMI monitor protocol

HMI monitor protocol instruction:

HMI monitor protocol is interior protocol in PLC.It used between Autoshop software and PLC for communication.Autoshop can erased,read and download user program through the protocol. HMI on sale support to H2U/H1U communication, also used this protocol. It can read ,write and control parameter, concretely can monitor state of any component in PLC , force modified any component, and control run/stop state of PLC etc..

Hardware and software config when use HMI monitor protocol:

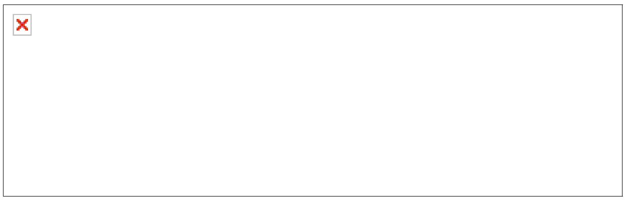
Baud rate and communication format of this protocol are fixup.

If set COM1 protocol is HMI monitor protocol or COM0 two line mode, erase,read and download program are unsupported.

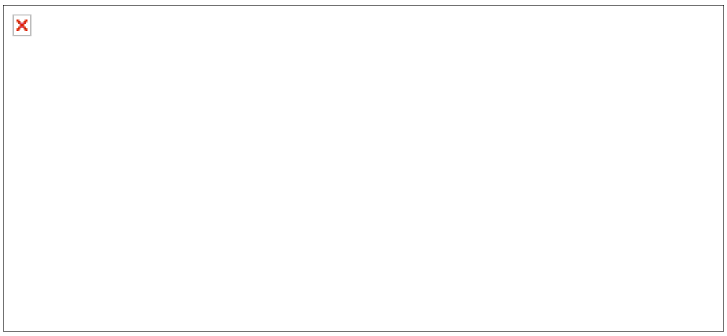
port	the number of communication line	JP0 setting	protocol config word	PLC state	protocol
COM0	RS422,MiniDIN8port	short	unconditional	STOP	HMI monitor protocol,can download and upload program
				RUN	
	RS485,2 line port	OFF	D8116=H01	STOP	HMI monitor protocol, can't download and upload program
				RUN	
COM1	H2U-422-BD extended port	short	D8126=H81	STOP	
				RUN	
	RS485,2 line port	short	D8126=H01	STOP	
				RUN	

parallel connection application

When data need to change between two series of H2U/H1U PLC main modules ,use 1:1 parallel connection protocol is one of the simplest types to communicate.You can use twisted-pair parallel connect corresponding RS485 signal port in COM1 port of two PLC to buildup communication network.



PLC system software set up parallel connection protocol inside.User only need set D8126 in system register.Set one PLC as parallel connection protocol host station and common communication mode, only require to sentence as follows:



Set another PLC as parallel connection protocol slave station and common communication mode,only require to sentence as follows:

Connect COM1 of two PLC,two PLC can auto exchange data. Data section address of communication commutative is fixed, receive and send corresponding fixed variable area of each other.according to data size of commuted,divide into two communication mode，table as follows:

	host station send(slave station receive)	slave station send(host station receive)
common mode M8162=0	M800~M899 D490~D499	M900~M999 D500~D509
high speed mode M8162=1	D490~D491	D500~D501

parallel connection protocol communication and correlative control variable are as follows:

M8070: setting 1 is parallel connection host station. If the bit is 0, D8126=50h is parallel connection host station too.(setting of M8070 take precedence of D8126's)

M8071: setting 1 is parallel connection slave station .If the bit is 0,D8126=05h is parallel connection host station too.(setting of M8071 take precedence of D8126's)

M8162: High speed parallel connection mode.

M8072: Parallel connection running

M8073: Parallel connection setting abnormity

M8063: Serial communication error

D8070: Setting error time for judged,default is 500.

D8063: Serial communication error code .

N:N online communication application

When one equipment have many(2~8) PLCs need commute information each other and running in phase,can use PLC built-in N:N network protocol,achieve polytropic communication among PLC.On hardware you only need use twisted-pair parallel connect all of PLC's RS485 signal port of COM1 port to compose to communication network.

User need set one of PLC is N:N protocol host station ,the value of host station number (D8176) is 0,the value of speed mode is D8176,for example:

Set other PLC are N:N protocol slave station, station number are D8176,speed mode are D8178,for example:

When PLC running,can be exchanged data among many PLCs.User program can read state data of other PLC send in interior specifically data section of local PLC. Host station user program copy the data need broadcast to specifically data cell，then other PLCs can read . According to two required guideline that information communicating need and communication refresh speed ,there are three modes can be choose,corresponding each variable area definition as follows:

N:N communication mode setting	station No.	device number	
		bit device(M)	word device(D)
Mode 0 D8178=0	No.0	Null	D0~D3
	No.1	Null	D10~D13
	No.2	Null	D20~D23

change data 0 M component 4 D component	No.3	Null	D30~D33
	No.4	Null	D40~D43
	No.5	Null	D50~D53
	No.6	Null	D60~D63
	No.7	Null	D70~D73
Mode 1 D8178=1 change data 32 M component 4 D component	No.0	M1000~M1031	D0~D3
	No.1	M1064~M1095	D10~D13
	No.2	M1128~M1159	D20~D23
	No.3	M1192~M1223	D30~D33
	No.4	M1256~M1287	D40~D43
	No.5	M1320~M1351	D50~D53
	No.6	M1384~M1415	D60~D63
	No.7	M1448~M1479	D70~D73
Mode 2 D8178=2 change data 64 M component 8 D component	No.0	M1000~M1063	D0~D7
	No.1	M1064~M1127	D10~D17
	No.2	M1128~M1191	D20~D27
	No.3	M1192~M1255	D30~D37
	No.4	M1256~M1319	D40~D47
	No.5	M1320~M1383	D50~D57
	No.6	M1384~M1447	D60~D67
	No.7	M1448~M1511	D70~D77

Instruction of how to Set N:N link protocol's register:

D8126: COM1 communication port communication protocol config,set to 40h means N:N host station;set to 04h means N:N salve station.

D8176: Station number, range from0 to 7,0 is host station.

D8177: The total number of Slave station, range from 1 to 7,only host station need set.

D8178: Refresh range（mode）setting,range from 0 to 2,only host station need set.

D8179: Setting Re-try times,only host station need.

D8180: Setting bound of communication time out ,unit is 10ms,only host station need.

M8183~M8190:sign of communication error,M8183 correspond to station number 0 （host station）,M8184 correspond to station number 1,the rest may be deduced by analogy,M8190 correspond to station number 7.

MODBUS protocol instruction

Basal layer of MODBUS communication is RS485 signal. It link up with twisted-pair, transmit distance so far that can reach 1000 meters, anti-jamming capability is good and low cost .In communication of industry control equipment,it is use abroad,so many manufacturers's transducers and controllers are use this protocol.

communication of data format have two type :HEX code data and ASCII code, respectively named MODBUS-RTU and MODBUS-ASC protocol,the former data communicate directly,but the latter communicate after switch data to ASCII code, so MODBUS-RTU protocol's communication utility, manage simply,used popularity.

MODBUS is single master mult-slave communication system,adopt master slave interrogator-responder system.Every time communication is initiated from master station, slave station passive responded.So controled equipment such as transducer,commonly inner install slave station protocol,and control equipment such as PLC,need provided with host station protocol¡¢slave station protocol.

Now MODBUS-RTU protocol set an example,explain typical format of communication frame:

Respond frame format: slave computer address+0x03+origin address of register +number of register+CRC parity

No.	data(byte)meaning	byte number	instruciton
0	head of frame	3.5 bytes leisure time	
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x03(operation code)	1 byte	read register
3	origin address of register	2 bytes	highbit ahead,lowbit after ,refer to register address
4	register number	2 byteS	highbit ahead,lowbit after(N)
5	CRC parity	2 bytes	highbit ahead,lowbit after
6	END	above 3.5 bytes leisure time	

Normal respond frame format:slave computer address+0x03+number of byte+value of register+CRC parity

No.	data(byte)meaning	number of byte	instruction
0	head of frame	3.5 bytes leisure time	
1	address of slave computer	1 byte	value range 1~247,set by D8121
2	0x03(function code)	1 byte	read register
3	byte number	1 byte	value: N*2
4	register value		every 2 bytes express one register value,highbit

		N*2 bytes	ahead,lowbit after .low.register address at the fore
5	CRC parity	2 bytes	highbit ahead,lowbit after
6	END	3.5 bytes leisure time	

If it is master station send communication frame error,or operate fault,send error respond frame,feedback to master station:

Error respond frame:slave computer address+(function code +0x80)+error code+CRC checkout.

No.	data(byte)meaning	number of byte	instruction
0	head of frame	3.5 bytes leisure time	
1	address of slave computer	1 byte	value range 1~247,set by D8121
2	function code+0x80	1 byte	error function code
3	error code	1 byte	1~4
4	CRC parity	2 bytes	highbit ahead,lowbit after
5	END	3.5 bytes leisure time	

Note:N of register number,max.is 1250 in H2U,max.is 50 in H1U.

When PLC programing,only need attention to information as follows:

Slave computer address:In master station send frame,the address shows target receive address of slave computer.In slave computer responsion frame, stand for master computer address;slave address's setting range is1~247, 0 is broadcast communication address.

Operation type: stand for W/R operation; 0x1=read loops operation; 0x03=read register operation; 0x05=write loops operation; 0x06=write register operation. For transducer,it only supports operation that read 0x03¡¢write 0x06.

Register origin address:Meaning register address that Slave computer called. For call on series of MD280,MD320 transducer,correspond to "function code number","command address","running parameter address"; Data number: Data number that call on "register origin address" in sequence, about register variable,unit is word.Register parameter(data): write data(host computer write),or read data(slave computer respond);

Check sum: CRC parity sum of current frame data, auto account in H2U/H1U, user needn't pay attention.

In communication process, hard to avoid communication error or fault , system software supplied special variable M8063,D8063 to report information of malfunction. If M8063 setting, means of appeared communication malfunction, user read content of D8063, acquired cause of malfunction.

In series of H2U and H1U PLC's system software,encapsulated MODBUS protocol, include MODBUS-RTU master station and slave station,MODBUS-ASC master station and slave station,used in COM1of communication port , use it only need setting corresponding data of system register D8126.

The communicational application of MODBUS master station

Communicational port COM1 in series of H2U and H1U PLC can used MODBUS-RTU and MODBUS-ASC instruction ,you only need set corresponding data of system register D8126.

Communication of MODBUS instruction (master station) have two type: RS expand instruction and MODBUS instruction,separately explained as following:

Used RS expand instruction achieved MODBUS communication program

Setting D8126 to H20, configured communicational protocol of com1 to MODBUS-RTU master station protocol. RS instruction communicated by MODBUS communicational protocol. In process of communication, engrossed register definition different from standard RS instruction, please pay attention to it:

In RS(MODBUS mode)expend instruction, each of operand definition different with standard RS instruction definition,separately:

is slave station address (high byte),communicational command(low byte,define by MODBUS protocol);

is register original address of call on slave station;

data length will be read or write,units is word;

is memory units original address for read or write data, engross continuous address units,length decided by n.

In RS(MODBUS mode)instruction, variable type that each of operand support are as following table:

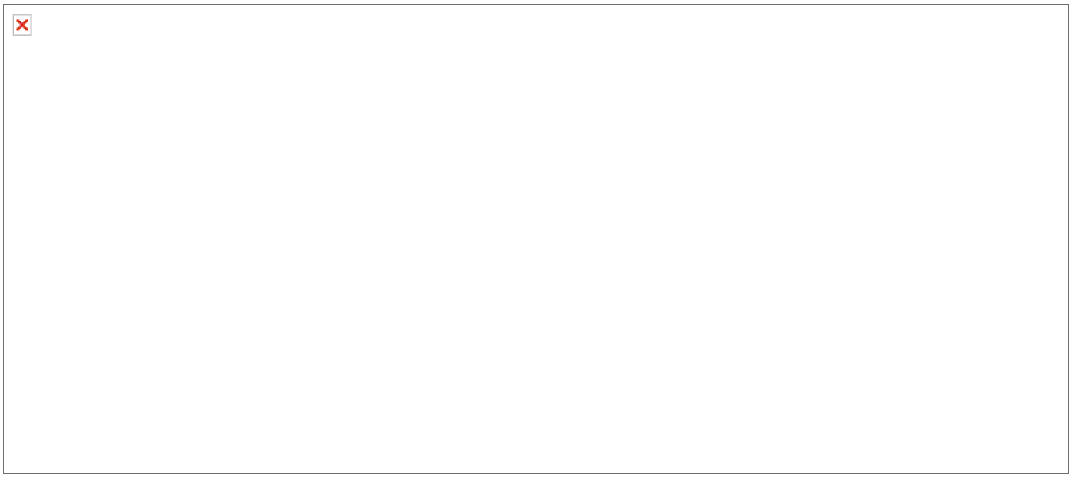
operand	word component										
	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
									✔		
	✔	✔							✔		
									✔		
									✔		

During coding,in front of every RS (MODBUS mode)instruction, evaluated finish every operand units according to communicational operation object address,operational type, operational register address,number of data,units of sent and received and so on. Once start executed,system program auto calculated CRC parity,organized communicational frame,accomplish operation of sent data,received responsion .

If communicated with MODBUS-ASC protocol(setting D8126 to H30),thereinto receive and send data's HEX-ASC format changing auto finish by PLC system program, the method user used RS(MODBUS mode) instruction and MODBUS-RTU protocol are all the same.

In the H2U,H1U program,If there were multi-RS (MODBUS mode) instructiond be drived,when system programs executed,yet after one RS instruction finished tache of "sent,waited answer,received,checkout parse stored",dealt with next RS instruction the same as,until all of RS instruction done with executed. When restart, user needn't cared about executive order and process,predigested PLC programme design,It wasMODBUS instruction's excellency of H2U.

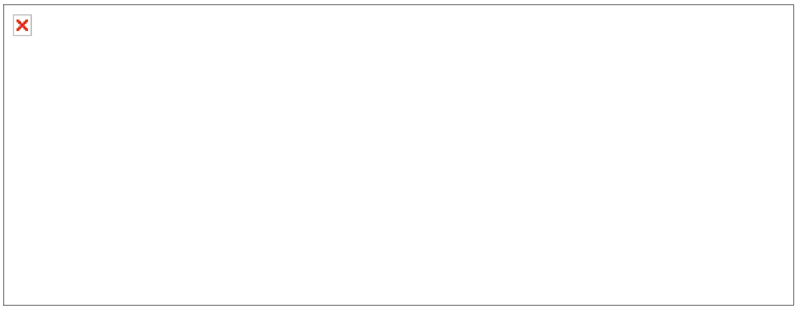
When RS (MODBUS mode) instruction finished one data sent,recieved answer operation, will auto reset M8123,used the sign,can judged whether RS instruction executed accomplish.programed reference as following:



Accomplished communicational programme by use MODBUS instruction

Setting D8126 to H20, configured communication protocol of COM1 to MODBUS-RTU master station protocol,upwards version V24120 in series of H2U of PLC,can communicated with "MODBUS" instruction directly.The data type of "MODBUS" instruction's 4 operands be supported were agilely,coding more convenience.

Same qualification as using RS expand instruction,you must set D8126 to H20(RTU) or H30(ASC)firstly,chose MODBUS master station protocol,then carried through MODBUS communication.



Thereinto operand:

is slave station address (high byte),communicational command (low byte,define by MODBUS protocol);

is register original address of call on slave station;

data length will be read or write,units is word;

is memory units original address for read or write data, engross continuous address units,length decided by .

In MODBUS instruction,variable type each operand supported as following table:

operand	word component									
	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
	✔	✔							✔		
	✔	✔							✔		
	✔	✔							✔		
									✔		

Compare RS (MODBUS mode) instruction with MODBUS instruction,latter operand variables all supported constant and D variable type,convenient for user coding.

Because one integrated RS (MODBUS) communication, finished with slave station's end of responsion, system program will reset M8123,after the instruction received tache finished .So user judge the instruction's end according as M8123.

In user's program,the RS (MODBUS) instruction circulational executed more less,and refresh of communicational data more frequentness,refresh speed of read data more quickly,advanced quality of real-time ,arrange read frequency of some unimportance parameter in reason, can improve effect of communication.

Using special variable M8129,can judged malfunction of communicational time-out, dealt with protect or give an alarm correspondingly.

The communicational application of MODBUS slave station

In some industry application, PLC controller be a part of industrial automatization system. To accept monitor of automatization control network ,typical Upper computer such as DCS,industrial PC of running group 2-state software etc., be monitor master computer, communicated with PLC equipment etc.

by MODBUS master station protocol ,now communicational port of PLC need communicated with Upper computer by MODBUS slave protocol. Series of H2U,H1U PLC installed MODBUS-RTU slave protocol and MODBUS-ASC slave station protocol, besides On COM0 and COM1 port can run the protocol.

The correlative register of MODBUS slave protocol

Communicational port	Setting word	Functionless definition	Instruction
COM0	D8116	protocol setting	02h:MODBUS RTU slave station 03h:MODBUS ASC slave station
	D8110	communicational farmat setting	
	D8111	COM0's salve station address of the PLC	Default is 1,modified availability when running
COM1	D8126	protocol setting	02h:MODBUS RTU slave station 03h:MODBUS ASC slave station
	D8120	communicational farmat setting	
	D8121	COM1's salve station address of the PLC	Default is 1,modified availability when running

In PLC program, after finished config of some registers above,When the communicational port in correspondence with had communicational frame that MODBUS master station sent to master computer address,PLC system program will auto organized MODBUS communicational frame responsion according to communicational request, needn't user program participated in.

Operation MODBUS slave station supported

When H2U/H1U be MODBUS slave station,communicational operational command supported MODBUS's 0x01,0x03,0x05,0x06,0x 0f ,0x10 etc. ,through this command,the variable that PLC's winding can be read-write were M,S,T,C,X(read-only),Y and so on ;register variable were D,T,C.

When master computer of MODBUS communication visited(read or write)interior variable of PLC slave computer,must followed definition of communicational command frame as follows ,as well as index method of variable address,then carried through normal communicational operation.

1.1 command code 0x01(01): read winding

Request frame format: slave computer address+0x01+winding original address+winding number +CRC parity

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x01(command code)	1 byte	read winding
3	winding original address	2 bytes	highbit ahead,lowbit after,refer to register address
4	winding number	2 bytes	highbit ahead,lowbit after(N)
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Response frame format£ºslave computer address+0x01+byte number +winding state +CRC parity

No.	Data(byte) meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x01(command code)	1 byte	read winding
3	byte number	1 byte	value: [(N+7)/8]
4	winding state	[(N+7)/8] bytes	Every 8 windings add up to a byte,if finally one less to 8 bits,un-definition part fill in 0.The 8 windings forwardly at first byte,the winding have littlest address at lowest bit.analogically in turn.
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Error response :refer to error response frame

1.2 Command code0x03(03):read register

Request frame format: slave computer address+0x03+register original address+register number+CRC parity

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247, set by D8121
2	0x03(commond code)	1 byte	read register
3	register original address	2 bytes	highbit ahead,lowbit after,refer to register address
4	register number	2 bytes	highbit ahead,lowbit after(N)
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Response frame format: slave computer address+0x03+byte number +register value +CRC parity

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x03(command code)	1 byte	read register
3	byte number	1 byte	value: N*2
4	register value	N*2 bytes	Eevery 2 bits indicated one register value,highbit ahead,lowbit after.which have less register address was at ahead
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Error response : refer to error response frame

1.3 Command code 0x05(05): write single winding

Request frame format: slave computer address+0x05+winding address +winding state +CRC parity

No.	Data(byte) meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x05(command code)	1 byte	write single winding
3	winding address	2 bytes	highbit ahead,lowbit after, refer to winding

			address
4	winding state	2 bytes	highbit ahead,lowbit after.availability except 0
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Response frame format: slave computer address+0x05+winding address +winding state +CRC parity

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x05(command code)	1 byte	write single winding
3	winding address	2 bytes	highbit ahead,lowbit after,refer to winding address
4	winding state	2 bytes	highbit ahead,lowbit after.availability except 0
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Error response: refer to error response frame

1.4 Command code 0x06(06): write single register

Request frame format: slave computer address+0x06+register address+register value+CRC parity

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x06(command code)	1 byte	write single register
3	register address	2 bytes	highbit ahead,lowbit after,refer to register address
4	register value	2 bytes	highbit ahead,lowbit after.availability except 0
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Response frame format: slave computer address+0x06+register address+register address+CRC parity

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121

2	0x06(command code)	1 byte	read single register
3	register address	2 bytes	highbit ahead,lowbit after,refer to register address
4	register value	2 bytes	highbit ahead,lowbit after, availability except 0
5	CRC parity	2 bytes	highbit ahead,lowbit after

Error response: refer to error response frame.

1.5 Command code 0x 0f (15): write multi-winding

Request frame format: slave computer address+0x 0f +winding original address+winding number+byte number+winding state +CRC parity

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x 0f (command code)	1 byte	write multi-single winding
3	winding original address	2 bytes	highbit ahead,lowbit after ,refer to winding address
4	winding number	2 bytes	highbit ahead,lowbit after.N,Max. is 1968
5	byte number	1 byte	value:[(N+7)/8]
6	winding state	[(N+7)/8] bytes	Every 8 windings add up to a byte,if finally one less to 8 bits,un-definition part fill in 0.The 8 windings forwardly at first byte,the winding have littlest address at lowest bit.analogically in turn.
7	CRC partiy	2 bytes	highbit ahead,lowbit after

Response frame format: slave computer address+0x 0f +winding original address+winding number+CRC partiy

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x 0f (command	1 byte	write multi-single winding

	code)		
3	winding original address	2 bytes	highbit ahead,lowbit after,refer to winding address
4	winding number	2 bytes	highbit ahead,lowbit after
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Error response: refer to error response frame.

1.6 Command code 0x10(16): write multi-registers

Request frame format: slave computer address+0x10+register original address+register number+byte number+register value+CRC partiy

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x10(command code)	1 byte	write multi-register
3	register original address	2 bytes	highbit ahead,lowbit after,refer to register address
4	register number	2 bytes	highbit ahead,lowbit after. N Max. is 120
5	byte number	1 byte	value:N*2
6	register value	N*2(N*4)	
7	CRC partiy	2 bytes	highbit ahead,lowbit after

Response frame format: slave computer address+0x10+register original address+number of register+CRC partiy

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	0x10(command code)	1 byte	write multi-register
3	register original address	2 bytes	highbit ahead,lowbit after,refer to register address
4	register number	2 bytes	highbit ahead,lowbit after.N,Max. is 120
5	CRC partiy	2 bytes	highbit ahead,lowbit after

Error response: refer to error response frame.

1.7 Error response frame

Elrror response: slave computer address+(command code +0x80)+error code+CRC partiy

No.	Data(byte)meaning	Number of byte	Instruction
1	slave computer address	1 byte	value range 1~247,set by D8121
2	command code+0x80	1 byte	error command code
3	error code	1 byte	1~4
4	CRC partiy	2 bytes	highbit ahead,lowbit after

2.1 Winding address

Winding: is bit variable£¬only have two states 0 and 1.The PLC includeM, S, T, C, X, Y variable etc. .

Variable name	Original address	Winding number	Instruction
M0~3071	0(0)	3072	
M8000~M8255	0x 1F 40(8000)	256	
S0~S999	0xE000(57344)	1000	
T0~T255	0xF000(61440)	256	
C0~C255	0xF400(62464)	256	
X0~X255	0xF800(63488)	256	
Y0~Y255	0xFC00(64512)	256	

2.2 Register address

Register: is 16 bits£¨word£©or 32 bits £¨double word£©variable,In the PLC,16 bits variable include D, T, C0~199;32 bits variable are C200~255.

Variable name	Original address	Winding number	Instruction
D0~D8255	0(0)	8256	
T0~T255	0xF000(61440)	256	
C0~C199	0xF400(62464)	200	
C200~C255	0xF700(63232)	56	32-bit register

Instruction: Through MODBUS visited 32 bits register that segment of C200~C255, one register as 2 register, one 32-bit register engross 2 16-bit register space.For example,user read or write 4 registers of C205~C208,MODBUS address is 0xF 70A (0xF700+10),register number is 8(4*2).

32 bits register un-support write single register(0x06) command code.

The programme method when PLCs communicate by MODBUS protocol

For 2 or more than 2 PLCs communicational parallel connection systems that programmed by MODBUS protocol ,had simple and flexible characters.It was more convenient in combinatorial system which had multiple equipments that PLC and MDI etc..

MODBUS communicational system was ---master and more slave mode. Exchanged data communication need was completely originated by host station. All of slave station were passive receiving and responding. The programme that correlative communication mostly run in host station program. In communicational program of slave station, only need configured communicational protocol,communicational format , station number of host computer and managed communicational data properly.

First exemple: For more PLCs communicational link as following,If achievable exchanged data between master PLC and # 2 slave PLC, as follows picture,exchange data:master(D50~D55,M10~M17)->slave(D100~D105,Y10~Y17) slave (D110~D119,X0~X17)->master (D60~D69,M100~M115).

Programme method:

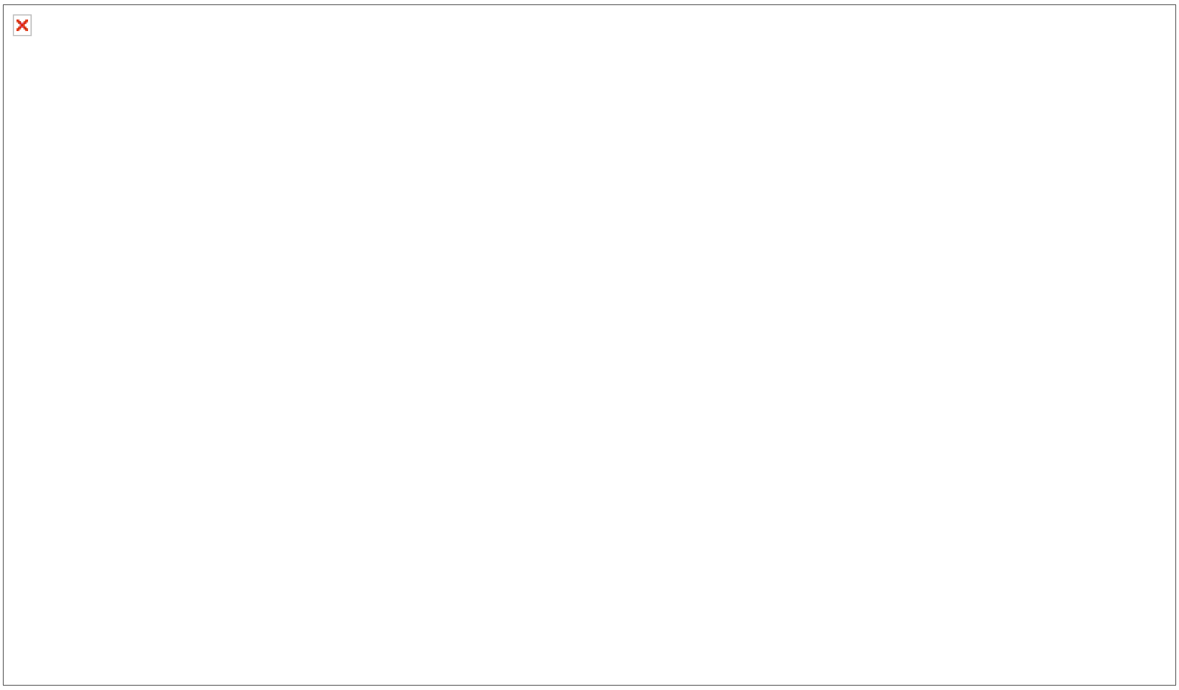
COM1 communicational port of master PLC configured to MODBUS host station protocol, 9600bps, 8N2 format, data's exchange (read-write) all achieved by master PLC.

Because there were part of bit variable of X,Y,M need data exchange,this variable be combined in D variable,In a piece seriate D variable area,exchanged groups by groups,each of master-slave both sides were combined and parsed of bit variable,the change with great efficiency,programmed simple.

M10~M17 variable in host station be combined in D56,the data master station sent were 7 D variable (D100~D106); X0~X17 variable in slave station be combined in D120,the data master station read were 11 D variable(D110~D120).

Host station programme illustration that used RS expand instruction:

Used host station program of MODBUS instruction(apply to version above V24120),finished same function of data change ,smelted sentence ,reduced engrossment of register:



#2 slave station's request of program were set COM1 port to MODBUS-RTU slave station,communicational format and host station the same, namely

9600bps,8N2,set host computer station number to 2,refreshed data register that host station read in time,host station maintained by the data wrote in communication ,Programme example as following:

Other programme method of slave station,refer to slave station's above.Notice: station number can't be set repeated.

Communicational programme method of PLC and a series of MD320 inverter

A series of MD inverters installed MODBUS-RTU slave station protocol. When PLC and MDI exchanged data by communicational mode,need communicated with MODBUS protocol. For mode system as following:

When coding£¬set COM1 port of PLC to MODBUS-RTU master station,9600bpsm, 8N2. In order to matched to leave factory default of a series of MD inverter, reduced bother of setting function code of inverter, PLC finished exchange work of data that system needed. Namely, slave computer(inverter default was #1 address)read-write command started or stoped control, set frequency, run parameter read etc. to inverter , or transmitted the set data of HMI to MDI.

Set COM0 port of PLC to(default) HMI monitor protocol, then PLC can communicated with HMI when run or stop state, programme conveniently downloaded.

COM0 port: protocol D8116= H01 ----download/ HMI monitor protocol

COM1 port: protocol D8126= H20; format D8120= H89;----modbus master station, 9600bps, 8N2

Communicational protocol of Inverter was MODBUS slave station,Its default address was ££1,9600bps,8N2.After initialized MD320, then setting like this. Passively respond exterior control(set FP-01=1 on inverter panel, when press ENT key ensured £¬then renewed default).

Notice when programming:

* Communicated and visited a series of MD inverter ,every communicational frame only can read or wrote a parameter, nonsupport series of address of multi-variable read-write in batch .namely number of register in communicational instruction n=K1.

* Function code of Inverter(address is HF***)can read repeatedly,but can't wrote repeatedly, be avoided to damage interior storage parts of an apparatus. Other parameters can read or wrote repeatedly.

Function code setting of MD320 cooperates with application of PLC

Main function code setting relation sketch map of MD320 inverter running control are as following, according to work mode needed,set indicatory function code in map,the function code can manual set by panel,can set by com communication also.

Setting of other a series of MD inverter's function code may be differently, but all of them have "start/stop command source" and "frequency command source" and so on basic function code setting, manage method can use for reference, please refer to corresponding use manual.

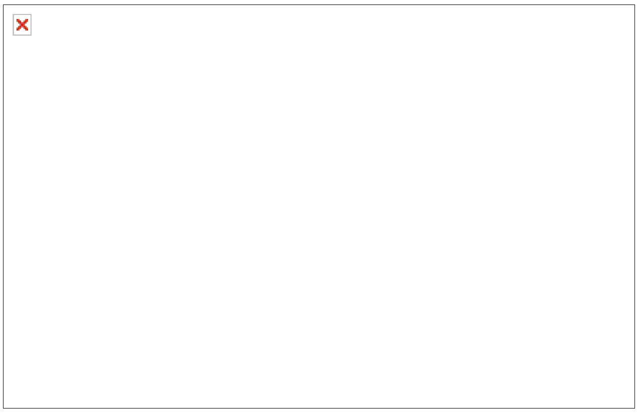
1) **Terminal start and stop,mult-segment speed control**

For connection as follows graph,according to requiring in graph define 4 logical import DI pins function in it :

Setting item	Setting function code	Instruction
Choose command source be "terminal"	F0-02 = 1	choose command source be "terminal"
"terminal command" mode is 2 line form	F4-11 = 0	be FRW/REV two terminals,default
According to actual connecting terminal, define its function	DI1: F4-00=1	"forward running"
	DI2: F4-01=2	"reverse running"
Choose frequency source be "mult-segment speed"	F0-03=6	Choose frequency source be "mult-segment speed"
According to actual mult-segment speed terminal, define its function	DI3:F4-02=12	mult-segment speed terminal 1
	DI4: F4-03=13	mult-segment speed terminal 2
Defined frequency of mult-segment speed used, notice default was 0.00Hz, evaluated in advance	FC-00=10.00Hz FC-01=15.00Hz FC-02=20.00Hz FC-03=30.00Hz	DI4/DI3=0/0: choose mult-segment speed 1(FC-00) DI4/DI3=0/1: choose mult-segment speed 2(FC-01) DI4/DI3=1/0: choose mult-segment speed 3(FC-02) DI4/DI3=1/1:choose mult-segment speed 4(FC-03)
Other	F0-07=0,default after initialize,only use main frequency source X.	

2) **Terminal start and stop, communication frequency control**

For connection as follows graph,according to requiring in graph define 2 logic import DI pins function in it,use inverter communication port and their Modbus-RTU slave station protocol:



Setting item	Setting function code	Instruction
Choose command source be "terminal"	F0-02 = 1	choose command source be "terminal"
"terminal command" mode is 2 line form	F4-11 = 0	be FRW/REV two terminals,default
According to actual connecting terminal, define its function	DI1: F4-00=1	"forward running"
	DI2: F4-01=2	"reverse running"
Choose frequency source be "com communication"	F0-03=9	Choose frequency source be "com communication"
Set communication same as Upper computer, suggest use default	FD-00=5 FD-01=0 FD-02=1 FD-05=1	"9600bps" Data format is "none parity" Local computer communication address (default is 1) Choose "standard Modbus" protocol
Running frequency modify at any moment	"H 1000" unit	Running frequency's percent value be write to H1000 unit with communication mode, (-10000~10000) correspond to (-100.00% ~100.00%) of maximum frequency
Other	FD-03/FD-04: communication responsion delay,manage communication timeout setting take the circumstances into consideration	

3)**Start and stop of communication, communicational frequency control**

For connection as follows graph,needn't use logical import DI pin function, use communication port of inverter:

Setting item	Setting function code	Instruction
Choose command source be "com communication"	F0-02=2	Choose command source be "com communication"
Choose frequency source be "communication port"	F0-03=9	Choose frequency source be "communication port"
Set communication same as Upper computer, suggest use default	FD-00=5 FD-01=0 FD-02=1 FD-05=1	"9600bps" Data format is "none parity",(Default 8N2 fixup) Local computer communication address (default is 1) Choose " standard Modbus" protocol
Start and stop control of inverter	"H 2000" unit	Write parameter separately: 1=forward running 2=reverse running 3=forward jog 4=reverse electromotion 5=free stop 6=decelerate stop 7=malfunction reset
Running frequency modify at any moment	"H 1000" unit	Running frequency's percent value be write to H1000 unit with communication mode, (-10000~10000) correspond to(-100.00% ~100.00%) of maximum frequency
Other	FD-03/FD-04: communication responsion delay,manage communication timeout setting take the circumstances into consideration	

4) **Run parameter of communication read inverter,modify function code**

As long as Upper computer use Modbus protocol and use same communication format config with MD320 inverter,can real time communication with inverter,read inverter parameter,even modify function code.

Read running parameter part:

parameteraddress	parameter description	read-write property
H1000	communication setting value(-10000~10000)	R/W
H1001	running frequency	R
H1002	generatrix voltage	
H1003	export voltage	
H1004	export electric current	
H1005	export power	
H1006	export torque	
H1007	running speed	
H1008	DI import sign	
H1009	DO export sign	
H100A	AI1 voltage	
H100B	AI2 voltage	
H100C	AI3 voltage	
H100D	count value import	
H100E	length value import	
H100F	thread speed	
H1010	PID setting	
H1011	PID feedback	
H1012	PLC process	

notice:communication setting value is relatively value's percent(-100.00%~100.00%),can do communication read-write operation.

Control command import to inverter:(write only)

command word address	command function
H2000	0001: forward running
	0002: reverse running
	0003: forward jog
	0004: reverse jog
	0005: free stop
	0006: decelerate stop
	0007: malfunction reset

Read the running state of inverter: (read-only)

parameteraddress	parameter description
H3000	1: forward
	2: reverse
	3: stop
	other:insignificance

Read the malfunction alarm code of inverter:(read-only)

Malfunction alarm code address	Malfunction information of inverter
H8000	0000: no malfunction 0001: inversion unit protect 0002: accelerated over electric current 0003: decelerated over electric current 0004: constant speed over electric current 0005: accelerated over voltage 0006: decelerated over voltage 0007: constant speed over voltage 0008: control power malfunction 0009: over low voltage malfunction 000A : inverter over loading 000B: motor over loading 000C : import lack direction 000D : export lack direction 000E : radiator over heat 000F : exterior malfunction 0010: communication malfunction 0011 : contactor malfunction 0012 : electric current detect malfunction

Description data of Communication malfunction information (malfunction code):

communication malfunction address	malfunction function description
8001	0000: no malfunction

	0001: password error 0002: command code error 0003: CRC parity error 0004: inefficacy address 0005: inefficacy parameter 0006: parameter modify inefficacy 0007: system be locked

Read or modify the inverter function code parameter:(read-write)

parameter address	parameter description
HF000	The parameter of inverter function code F0-00
HF001	The parameter of inverter function code F0-01
...	
HF711	The parameter of inverter function code F7-17
HFB1E	The parameter of inverter function code FB-30

When read or modify inverter function code,"register address" is "function code" number.If read F0-01 function code,"register address" is HF001,express by Hex format, thereinto high byte is function code group number,low byte is function code index number in group,pay attention to the index number need Hex format.for example need to read no. FB-29 function code,"register address" is HFB1D,the rest may be deduced by analogy.

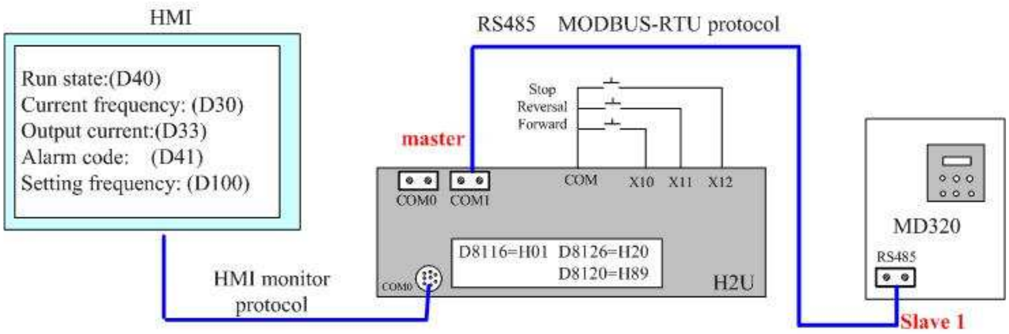
Function code of inverter have their modify property limit,some function code can be modified at any moment,other only can be modified at stop state,range of some modified value restrict by other correlative function code etc.,need to pay attention.

Every MODBUS communication frame of a series of MD320 inverter only can read one Word parameter,nonsupport a communication frame visit to series mult-address.

When need random rewrite to function code(HF***),and don't effect to inverter memorizer's life,you can replace the address by H0*** ,namely replace '0' to 'F' which is high 8 bit of function code address,low 3 hex bit not change. The modification take effect immediately,but write to memorizer at inverter be wrote and power cut.

Programme of PLC modified MD320 function code

T2U--MDI connects with communication mode,executes operation that start,stop,forward,reversion and runs frequency control of inverter by H2U,real time reads running parameter of inverter. For example system as follows:



Required function as follows:

1) When everytime electrify running, H2U auto sets command source (F0-02=2) of inverter to serial port, don't send modified command after modification finished; It is independent of electrify order of two components H2U and MD320;

2) When <![endif]> everytime electrify running, H2U auto sets frequency choose £¨F0-03= 9) of inverter to serial port, don't send modified command after modification finished;It is independent of electrify order of two components H2U and MD320;

3) <![endif]> Press button X10, inverter runs forward ; press button X11, inverterruns reverse; press butter X12, inverter stop running;

4) Frequency value of D100 as run frequency of inverter,circularly send to inverter.

5)Circulative read currently parameter that run state,running frequency,output current,power and so on of inverter.

Explanation of PLC program essential:

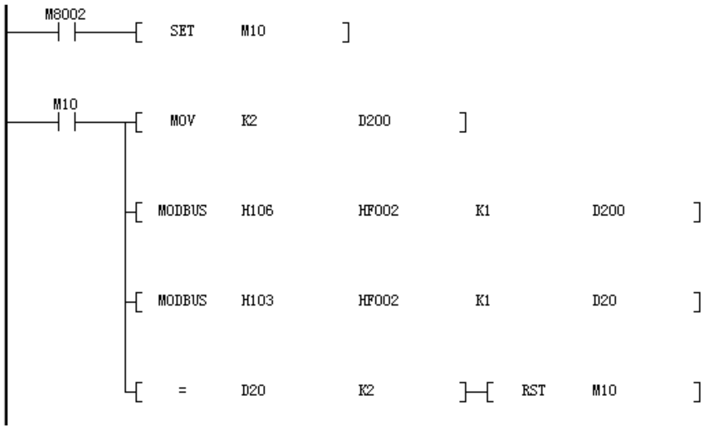
D8116=H01, sets COM0 to download and HMI monitor protocol;

D8126=H20,D8120=H89, sets COM1to MODBUS-RTU protocol, 9600bps, 8N2, program to operation of COM1 by MODBUS instruction or RS extend instruction.

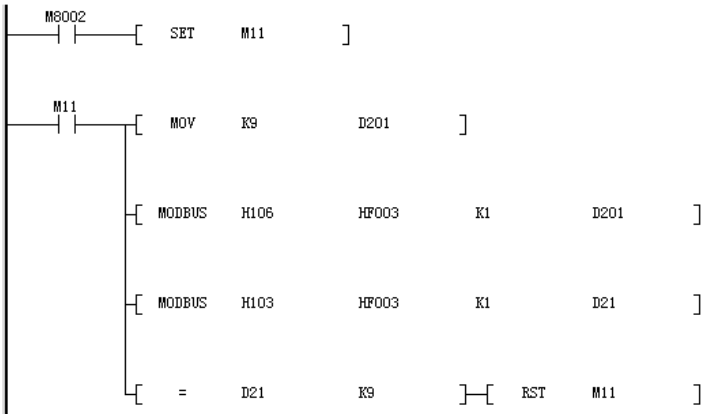
1)Sentence illustration about MODBUS protocol setting (You had better set on first line position of PLC program)

2)Sentence illustration about read operation of function code:

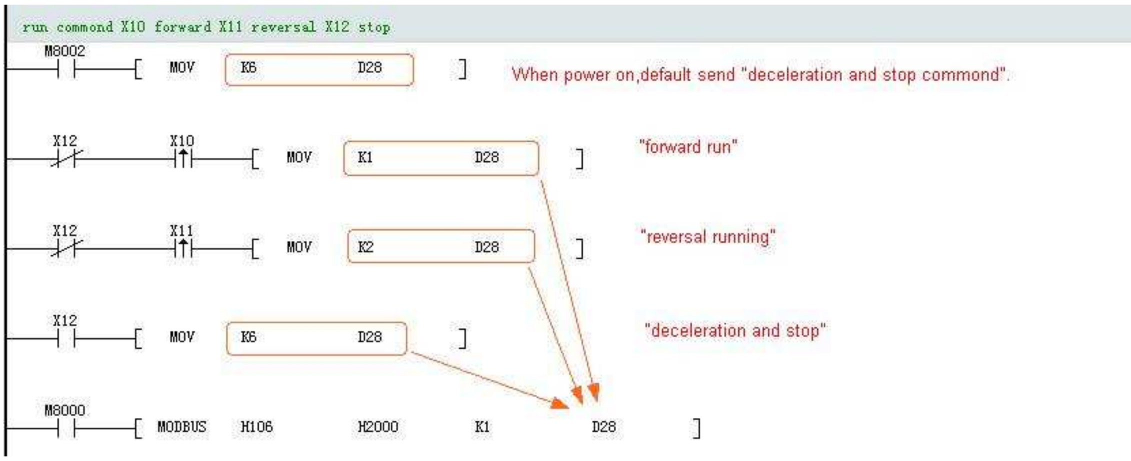
Modified " command source" function code (F0-02) to "communication start/stop ", namely F0-02=2:



Modifies " frequency source"function code (F0-03) to "communication given", namely F0-03=9:



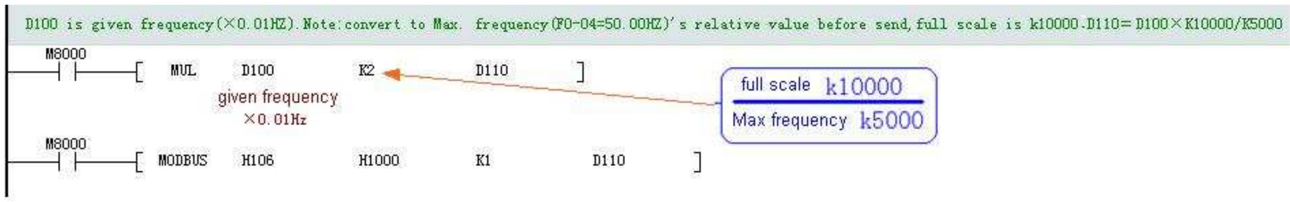
3)Command response to button, send forward run, reverse run or stop instruction of inverter ,"STOP"is prior:



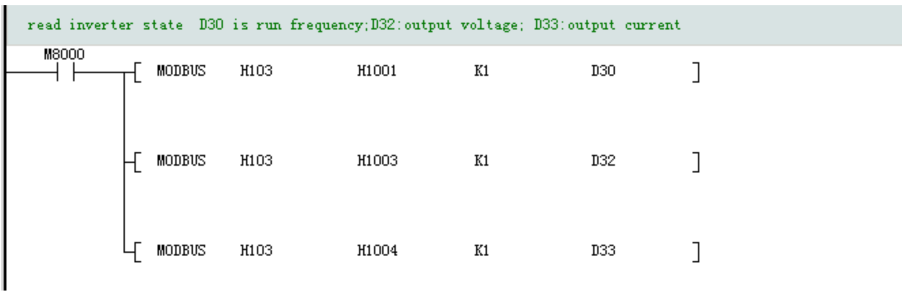
Deal with command word of three operation response, send by one and the same MODBUS or RS instruction.

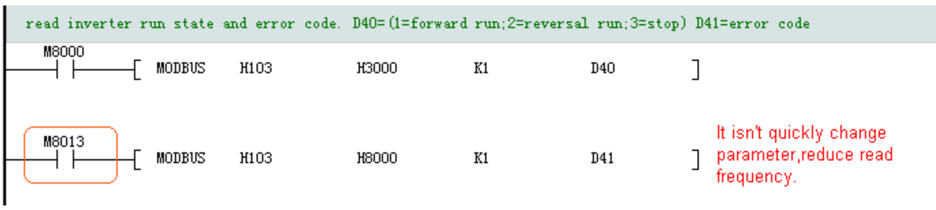
4)Changes running frequency of inverter:

The frequency instruction send to inverter, isn't data which dimension is 0.01Hz, but percent value relative to " maximal frequency", K10000 is full graduation, before sending it need converts, for example inverter maximal frequency is 50.00Hz,runs by 40.00Hz, send data needed is 40.00×K10000/50.00=K8000. In the example,directly replaces k10000/k5000 by K2. In actually programme,if maximal frequency isn't 50.00Hz,you had best faithfully calculates by circularly send instruction:



5)For circulative read operation illustration of running parameter:





The communication of CAN

CAN network is popular in industrial user for more higher communicational velocity than RS485,capability of anti-jamming good, simply wiring.A series of H2U PLC which are N, XP mode,a series of H1U PLC,after CAN expand card installed,It supported communication based on CAN network,supplied the CAN freely communication sending and receiving instruction of CANTX,CANRX and so on.user according to communication protocol organise send or receive operation of communication frame by oneself.

In CAN network,no difference of master and slave,allowed polytropic communication .If appeared communicational collision phenomenon,priority arbiter of CAN's base course communication to made station have high PRI sent priority .

Picture as follows respective are H1U-CAN-BD,H2U-CAN-BD and their function instruction picture.In two picture, same grade indicative is same function point.

1. Error pilot lamp (red),corresponding silk-screen mark is "ERR", when communication error,lamp lighted.

2. communications pilot lamp (Kelly),corresponding silk-screen mark is "COM",In unit time, communication data volume more bigger,pilot lamp twinkle more continually,pilot lamp put out when nocommunication.

3. power pilot lamp (Kelly),interior logical power indication (see instruction on table 1),when main module electrified pilot lamp lighted.

4. Interior port through the port exchanged data with main module.Interior logic power also through the port supplied to communications card by main module.

5. Dial code switch,use to set local computer address,communication baud rate,data line matching resistance connection.

6. Receive data pilot lamp(Kelly),corresponding silk-screen mark is "RXD",when received data,the pilot lamp twinkle.

7. Bus port, also named user port, the description of function refer to table 1. Serial numberin circle means pin number.

8. Fixup CAN card 's bolt hole.

9. send data pilot lamp(Kelly),corresponding silk-screen mark is "TXD",when sending and receiving data, pilot lamp twinkle.

Definition of Bus port pin:

Pin number	Signal	Description
1	+24Vcc	EXT DC 24V power supply positive IN
2	CANH	CAN bus positive
3	PGND	Shielding earth wire,contact to communication cable shielding layer.
4	CANL	CAN bus negative
5	0V	EXT DC 24V power supply negative IN

When composed to CAN network, the upwards five lines (included shield) of all the equipment,all need------connected up one by one.And +24Vcc and 0V needed ext 24V DC power .Both ends of bus needed to add 120ohm 's CAN bus matching resistance.

There has a 8-bit dial code switch on CAN communication card,be used to set station number of module,chose baud rate,whether terminated matching resistance. as follows picture,dial code switch have SN every bit, "ON" means logic " 1"

CAN-LINK dial code switch function definition of every bit.

Dial code number	Signal	Description
1	address wire A1	The 6 bit dial code switch from high to low compound to 6 bit binary digit,sign local computer station number(any more you can set station number by D component),"ON" means 1,"OFF" means 0. combining with above mode: A 6A 5A 4A 3A 2A 1. Such as both of A5,A4 are ON,other bit is OFF,binary address is 011000,hex is h18,change to decimal is K24,stand for CAN address station number of local computer is # 24.
2	address wire A2	
3	address wire A3	
4	address wire A4	
5	address wire A5	
6	address wire A6	
7	baud rate	OFF:high speed mode,baud rate is 500Kbps,ON:low speed mode,baud rate is 100Kbps
8	matching resistance	If dial code switch is ON,means access in 120 ohm's terminal matching resistance,otherwise OFF

The communicational instruction of CAN

Send data instruction format of CAN: **CANTX**

Receive data instruction format of CAN: **CANRX**

Definition of Each of parameters are as follows:

, two parameters together composed CAN address,the definition method of two address same as address definition of CAN communication protocol. If equal to " 0" means standard CAN address(11bit), bit0~bit10 of means address. If bit13 of equal to 1,means expand address mode , low 29 bit address of and together compose of CAN address.

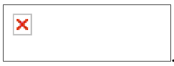
In CANTX instruction, is send buffer. In CANRX instruction,it is receive buffer. Max. 4 D components Begin from the D component be send or receive buffer.

In CANTX instruction, is number of send data, In CANRX instruction,it is number of receive data .Unit is byte,Max. is 8.

In CANTX/CANRX instruction, data type corresponding operand allowed as following:

Operand	Word component									
	K	H	KnX	KnY	KnM	KnS	T	C	D	V	Z
	✔	✔							✔		
	✔	✔							✔		
									✔		
	✔	✔							✔		

A CANTX/CANRX instruction allowed send/receive 1~8 byte, for example the instruction as follows:

Achievable operation:sent 3byte data that begin to host computer D100 to station of N0.#23 address. Pay attention to order of data send: .

If n= k8, D100 = h1234, D101 = h5678,D102 = h9ABC, D103 = hDEF0

Data of send as follows table:

No.1 byte	No.2 byte	No.3 byte	No.4 byte	No.5 byte	No.6 byte	No.7 byte	No.8 byte
h12	h34	h56	h78	h 9A	hBC	hDE	hF0

For receive data storable order of CANRX instruction also same as this ,If receive port byte from high to low, CAN receive data register MDL = h12345678, MDH = h9ABCDEF0

If n = 1,Only send 1 byte:h12

If n = 3,send forward 3 bytes: h12,h34,h56.

Analogize in turn.

In program, write multi-CANTX/CANRX instruction can exchange more data, and need not time-sharing drive. You can use M8000 drive CANRX receive instruction at the same time, but can only receive of CANRX one by one. Only first item receive finished or received timeout, can execute next instruction receiving, execute in order,the time-sharing PLC auto manage on background, user need not program and time-sharing. If send instruction and receive instruction's ID can't matching, data will lose.

When PLC system software meet CANTX instruction,will startup execute immediately, firstly send data frame , then wait reply by destination station, until finish, will execute next program sentence.You can judge whether executing send successful by special register M8194.

M8194: ON=CAN send fault; OFF=send successful

When PLC system software meet CANRX instruction,will startup execute immediately, turn to CAN receiving and waiting state, PLC will execute other program sentence at waiting and receiving state.When next circulation executed the sentence,deal with receive data judgement .It will not start execute next CANRX instruction, until a CANRX instruction finished.

But whether receive data, didn't dicide by local station, also lay on whether other equipment had data to send.If waiting time exceeded the time D8241 setting , timeout error, M8192 resetting. If met an availability drivable CANRX instruction after,system software would start execute of next CANRX instruction.

When use multi-CANRX instruction, check whether a sentence is executed succssful, can judge by special register M8193 correlate to CAN communication:ON=CAN receiving ;OFF=received finish, leisure state.

Instruction example 1:

10ms sent a group data,buffer is D100~103,D110 stored send byte

If D110 = k8, D100 = h1234, D101 = h5678, D102 =h 9ABC, D103 = hDEF0

Send data as following table:

No.1 byte	No.2 byte	No.3 byte	No.4 byte	No.5 byte	No.6 byte	No.7 byte	No.8 byte
h12	h34	h56	h78	h 9A	hBC	hDE	hF0

If receive port byte for high to low,CAN receive data register MDL =h12345678, MDH = h9ABCDEF0

If D110 = k1, only send a byte:h12

If D110 = k3, before send 3 byte: h12, h34, h56, Analogize in turn.

Instruction example 2:('s bit13 is 0)

Supposed 8 PLCs linked to CAN network,be wrote the send instruction in one PLC program as follows

The PLC send data that in D10~D13 register to address H200, because CAN protocol don't care about master-slave station,so data that the PLC sent to address H200 is exoterically. Any PLC in network that want to receive D10~D13 data of this PLC,can write receiving instruction to any PLC program as follows :

Only executed the sentence above,you can receive address that in H200 and store the data in D100~D103. You can write program in many PLCs to receive this data.

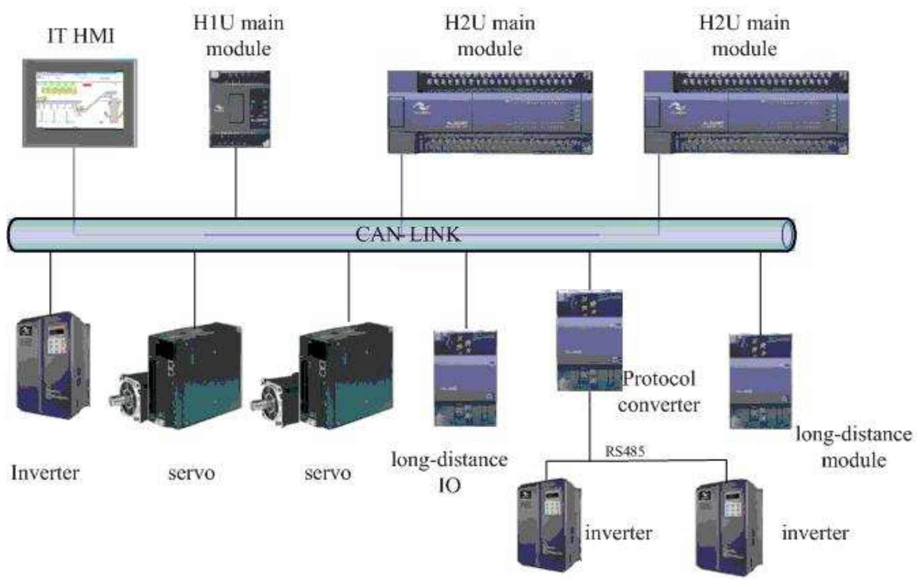
Samely,any PLC want to send data. can also write CANTX instruction to program, 's address freely defined by user (Notice:according to definition rule of 11bit identifier and 29bit identifier),as long as the receive one write same address in CANRX instruction can be receive data that the user defined address.

Instruction example 3:('s bit13 is 1)

Now is H 334F(binary system 11,0011,0100,1111), is H200(0000,0010,0000,0000). 's bit13 is 1, means 29bit address.The address compose of (lowness 16 bit address)and 's bit0~bit12(highness 13 bit address),namely 's 11,0011,0100,1111 and compose address 11, 0011, 0100, 1111, 0000, 0010, 0000, 0000(hexadecimal H 334F0200). I.e. PLC send data in D100~D103 register to address H 334F0200 .

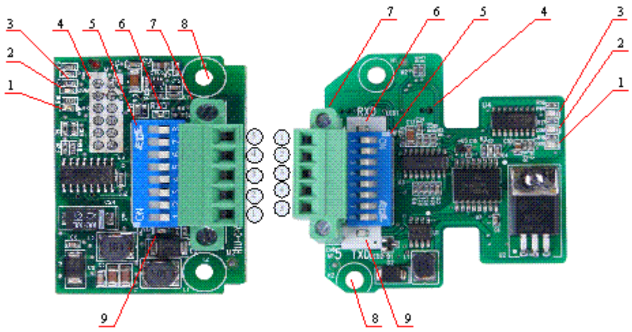
The communication of CAN-LINK

A series of H2U of the N,XP model PLC,a series of H1U of PLC, After installed CAN expand card, be able to support communication that base on CAN network, INOVANCE base on CAN communication network and protocol , defined a specifically CAN communication protocol,CAN communication equipment that base on the specifically protocol can be connected directly .The network named CAN-LINK network. Now many control and transmission equipment of INOVANCE,such as a series of IT man-machine interface,a series of H2U/H1U PLC,a series of MD inverter,a series of IS servo driver etc. ,all supported to CAN-LINK communication. The compose of network was as following graph:

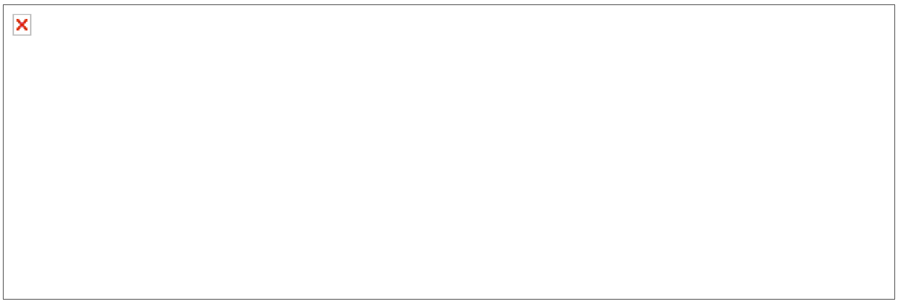


In PLC system software, communication base on CAN-LINK protocol, the embody operation to each of tache of sending and receiving,encapsulated software function ,adopted FROM/TO instruction, can CAN-LINK communication,visited this CAN-LINK exterior equipment, simplely as visit expand module.

Form of CAN communication expand card as follows:



Electrified N model,XP model of H2U PLC master module , when detected H2U-CAN-BD expand card,would auto initialized CAN communication port .Electrify a series of H1U PLC master module ,when detected H1U-CAN-BD expand card,would also auto initialized CAN communication port ,got ready to CAN protocol,or CAN-LINK protocol communication.



Remark:recommended CAN communicate cable , manufacturer: ShenZhen LianJiaxiang science and technology Ltd. model: RVVP 2×2×0.5.

When FROM/TO instruction of PLC visited local expand module, module address number must be #0~#7. If module address number that FROM/TO instruction visited bigger or equal to # 100, defined through CAN-LINK network visited the exterior CAN station equipment. For example,when the number visited module address is # 120, means peripheral equipment that visited with CAN-LINK network #(120－100)=No. # 20 CAN address.

Definition of H2U-CAN-BD expand card setting switch be the same with CAN free communication, be the same with CAN-LINK communication too.

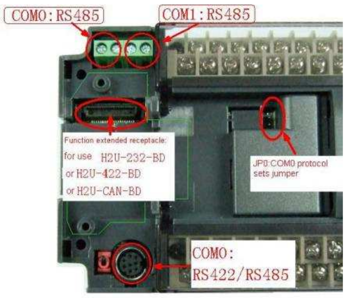
The equipment that the CAN-LINK protocol or the long-distance expand module visited protocol. Through the expand module FROM/TO instruction,can read-write long-distance expand module (need expand module suppprted) and long-distance PLC that linked by CAN.

In the free CAN protocol,needn't assigned station number of PLC. The equipment that the CAN-LINK protocol or the long-distance expand module visited protocol, need assigned station number of each of PLC or the long-distance expand module. Particular use please reference to <appendix 5.13 CAN communication instruction>.

H2U-HMI communication link

Signal link of H2U

The port of H2U have Mini DIN8 signal electrical outlet communicate with HMI. The electrical outlet in common use download/HMI monitor of user program. And RS485 communicational port of bolt fixation base,for convenience using twisted-pair communicational application link. The two group ports as long as configured properly,can communicated with HMI for COM0.



Instruction:

1)The communicational protocol, full duplex or semiduplex mode that communicational port used was decided by JP0 and user program:JP0 is OFF (plug in jumper wire cap),COM0 set to RS422 mode ,can download PLC user program. JP0 is OFF(pull down jumper wire cap),work mode of COM0 decided by D8116, D8116=H01 meant RS485(2W),D8116=H81meant RS422(4W).

2)This Mini DIN8 signal cable was compatible with model of SC-09 download cable (DB9 pin RS232---transformable cable of Mini DIN8 pin RS422) that sell on market .If HMI standard config was DB9 electrical outlet and used cable link mode of Mini DIN8 electrical outlet, suggested using RS422 full duplex communicational mode.

Signal link of H1U

The port of H1U have Mini DIN8 signal electrical outlet (COM0) communicate with HMI. The electrical outlet in common use download/HMI monitor of user program. And RS485 communicational port of bolt fixation base,for convenience using twisted-pair communication link.

Instruction:

1)If HMI standard config was DB9 electrical outlet and used cable link mode of Mini DIN8 electrical outlet, suggested using RS422 full duplex communicational mode.

2)This Mini DIN8 signal cable can used model of SC-09 download cable(DB9 pin RS232---transformable cable of Mini DIN8 pin RS422) that sell on market .

HMI supplied RS485 semiduplex communicational protocol, strengthened conveniency of user using HMI-PLC communication link ,suggested manufacturer of HMI supported RS485 semiduplex communicational mode.

Communication config of HMI and H2U

HMI and H2U,H1U communication connecte with RS422 (4W) or RS485(2W)level .

Communicational baud rate fixup as 9600bps,7E1 format, namely data bit is 7-bit, even, stop bit is 1 bit.

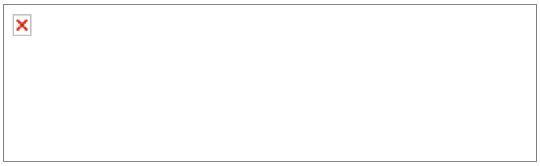
When HMI programme,user only need to choose type of"INOVANCE H2U" PLC slave machine,auto set and communicate with the config above. pay attention to accord with connection mode actual used, both of HMI and H2U configure four line (4W) or two line (2W) mode .

H2U-HMI use the method of RS422 full duplex(4W)

Project 1: MiniDIN8 receptacle of COM0 communication port:

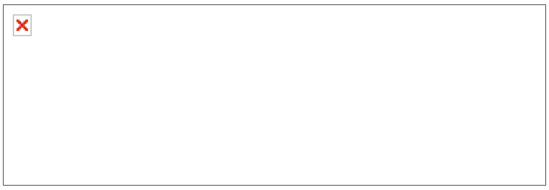
Project 2: COM1 communication port, need increase H2U-422-BD expand card:

Note: In PLC program, choose instruction of communication protocol must evaluate to special register of D8126 in frist run period.When PLC runs, modify the register is invalid.

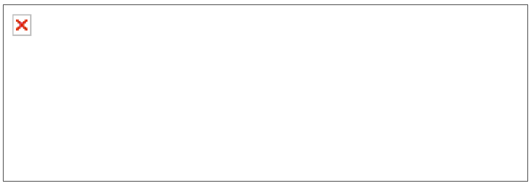


H2U-HMI use the method of RS485 semiduplex(2W)

Project 1: COM0 communication port's RS485+ and RS485- connection terminal:



Project 2: COM1 communication port's RS485+ and RS485- connection terminal:



Note: In PLC program, choose instruction of communication protocol must evaluate to special register(H01) of D8116 and D8126 in frist run period.When PLC runs, modify the register is invalid.

H1U-HMI use the method of RS422 full duplex(4W)

MiniDIN8 receptacle of COM0 communications port:

Note: COM1 port can't carry out communication connection of RS422 full duplex(4W),but use RS485 semiduplex(2W) mode.

H1U-HMI use the method of RS485 semiduplex(2W)

Project 1:COM0 communication port's RS485+ and RS485- connection terminal:

Project 2:COM1 communication port's RS485+ and RS485- connection terminal:

Note: In PLC program, choose instruction of communication protocol must evaluate to special register(H01) of D8126 in frist run period.When PLC runs, modify the register is invalid.

Difference in H1U and H2U device

Difference in H1U and H2U just are config of M variable:

Device	H2U	H1U
M component	3072 M variables ,namely M0~M3071	1356 M variables,namely M0~M1535,inexistence M1356~M3071
C,D,T,S	same	
X,Y	Number of H1U decide on material mode of main module	

Communication protocol of HMI configure to H1U same as H2U, only range of M variable address less than.

So,you can choose model of protocol as follows,when there is not supply protocol option "Inovacne H1U"in program environment of HMI:

"**Inovacne H2U**" protocol , or " **FX2N**" protocol.

H2U-4AD H2U-4ADR 4 channels simulation import module

4AD(R)extended module cooperated a series of H2U main module in work,to accomplish signal detected of 4 simulation import channels,transformed signal which range was -10V~10V or -20mA~20mA to 12bit digital, read by PLC main module. main module visited BFM unit of register in local extended module by FROM/TO instruction .

Electric specification

Item	Guideline	Instruction
voltage import signal level	-10~10V DC	Every channel can choose voltage signal type by oneself.
Voltage channel import impedance	200KO	
Electricity import signal	-20mA~20mA,	
Electricity import sampling resistance	250O	
Number of import channel	4 channels	
Import signal frequency	less than 10Hz	
Transform speed	15ms/channel（constant speed），6ms/channel（quickest）	
Digital export	12bit: -2048~+2047	
Dpi	voltage import 5mV, or electricity import 20uA	
Precision	±1% whole range	
Engross I/O count out	none	

Instruction of LED state indicator light:

1) Instruction of local extended module LED state indicator light

Item	Instruction
PWR	power supply of module digital circuit normal
24V	when exterior 24V power supply normal,it be lighted
COM	be lighted showed FROM/TO instruction visited module

2) Instruction of long-distance extended module LED state indicator light

Item	Instruction
PWR	module 24V power supply normal
COM	be lighted showed module communicating
ERR	be lighted showed there had error

3)Address serial number of local special extanded module:

Umbrella name of every extanded module（such as 4AD/4DA/4TC/CC-Link etc. module）except IO extanded is special module. PLC main module electrified everytime,will auto checked once all of extanded modules that connected ,and separately assigned "serial number" to special module and IO extanded port，user could not intervene or modify the result of their serial number，unless change module connect order.

The address serial number method that main module to special module began from main PLC module nearly,serial number were ＃0、＃1、…#7etc. in turn, didn't affect by middle inserted IO extanded module.

4)Address serial number of long-distance special extanded module:

For long-distance extanded module, value range is module communication station number +100,allow 62 long-distance extanded modules at most, through dial the dial code switch on the Station NO. , can set main module station number.

If A5,A4 of address dial code switch are ON,other are OFF,namely binary address is 011000,decimalist is K24. then we programme by FORM/TO instruction ,the module serial number is K24+100,namely is #124.

pay attention to CAN address's uniqueness,can't have same address.

5)Visited BFM section of 4AD(R)module:

PLC main module use method that read register buffer unit (BFM section) of 4AD (R)module, read digitization AD transformation result, setting module state by rewrite special BFM section. PLC main module visit the BFM section by read-write instruction FROM/TO.

There have EEPROM memory cell in extanded module, use to save same setting value of BFM, such as every simulation import channel 's signal type、offset 、gain etc.,save action of the cell is auto finished by setting state decision of corresponding BFM section.

Each register width of BFM section is 16bit (1Word),definition according to BFM section of 4AD(R) module as following table：

BFM	R/W	content
#0(E)	WR	choose channel signal mode, setting by 4 HEX bit,each

		HEX bit signify 1 import channel,highest bit is ch4，lowest bit is ch1: (default = H0000）,setting definition of each bit are as follows : 0=-10V~10V; corresponding digital export:-2000~2000 1=4mA~20mA; corresponding digital export: 0~1000 2=-20mA~20mA;corresponding digital export: -1000~1000 3=local channel off 4=-10V~10V; corresponding digital export:-10000~10000 5=4mA~20mA; corresponding digital export:0~10000 6=-20mA~20mA; corresponding digital export:-10000~10000	
#1	WR	channel1	Average filtering constant is number of sampling value that use for average calculation,setting range is 1~4096, default is 8. If need high speed sampling, set it to 1. When BFM#15 changed,auto resume to default .
#2	WR	channel2	
#3	WR	channel3	
#4	WR	channel4	
#5	R	channel1	The data that import channel sampling value after average filtering.
#6	R	channel2	
#7	R	channel3	
#8	R	channel4	
#9	R	channel1	the data import channel currently sampling,namely instantaneous value have no filtering
#10	R	channel2	
#11	R	channel3	
#12	R	channel4	
#13~14	-		reserved
#15	WR	choose ADC velocity	0=normal speed,15ms/channel(default) 1=speediness transform, 6ms/channel 1000~30000=high speed sampling,corresponding 1ms~30ms/channel
(＃16~19)	-	reserved	

<table>
<tr><td>#20(E)</td><td>WR</td><td colspan="9">1=reset setting parameter to default（leave factory value）.default=0</td></tr>
<tr><td>#21(E)</td><td>WR</td><td colspan="9">2=forbid to adjust offset/gain,（default）
1=allow to adjust offset/gain</td></tr>
<tr><td rowspan="2">#22(E)</td><td rowspan="2">WR</td><td>low 8bit correspond to 4 channels operation</td><td>G4</td><td>O4</td><td>G3</td><td>O3</td><td>G2</td><td>O2</td><td>G1</td><td>O1</td></tr>
<tr><td colspan="9">Offset/gain adjust enable，when non-zero,module write value of BFM23/24 to it interior corresponding channel control register, default=H00</td></tr>
<tr><td>#23(E)</td><td>WR</td><td colspan="9">Simulation import value （0、1、2 mode）that when offset,digital export is 0 ,default is 0.</td></tr>
<tr><td>#24(E)</td><td>WR</td><td colspan="9">Simulation import value（0、1、2 mode）that when gain,digital export,default is 5000.</td></tr>
<tr><td>#25~26</td><td>-</td><td colspan="9">reserved</td></tr>
<tr><td>#27</td><td>R</td><td colspan="9">4AD module software version</td></tr>
<tr><td>#28</td><td>-</td><td colspan="9">reserved</td></tr>
<tr><td>#29</td><td>R</td><td colspan="9">error state</td></tr>
<tr><td>#30</td><td>R</td><td colspan="9">Extanded module ID code，H2U-4AD's ID code is K2010.</td></tr>
<tr><td>#31</td><td>-</td><td colspan="9">reserved,inaccessible</td></tr>
</table>

State information which the meaning of the word BFM # 29 as follows:

BFM#29 bit number	ON status	OFF status
b0:	Errors. Either in b0 ~ b3a non-0, A / D conversion stops	no errors
b1:	Module EEPROM offset / gain setting is wrong	Offset / gain data correct
b2:	(impossibility)	power supply normal
b3:	module hardware malfunction	hardware normal
b10:	Digital output exceeds the range of -2048 ~ 2047	Digital output value normal
b11:	Sampling filter constant beyond the range of 1 ~ 4096	Sampling filter constant normal
b12:	Prohibition of the value of BFM # 21 set K2	permit BFM#21=K2
The other bit4 ~ 7, bit13 ~ 15 of BFM # 29 and so is not defined.		

BFM unit "(E)" in table is the project stored in EEPROM, with a property of power-down to maintain.

Register rewrite description:

Rewrite BFM # 0, # 23, # 24, etc. with the (E) unit trigger the operation to write EEPROM within the module. And write operation takes some time. Word will take about 300ms each time. So pay attention to PLC programming, when with the need to rewrite the multiple (E) BFM units, the user program delay for some time after to write a BFM unit above each time, do not write continuously to ensure that written instructions correctly.

Note: external 24V power supply of local expansion module failure. The system flag M8067 of PLC main module is set,error code D8067=k6708. Programming can regularly check the sign, and can find the error.

6)**Some explain of BFM section**

BFM#0 channel selection

Channel initialization. The 4 default channels are -10V ~ 10V, controlled by hexadecimal HXXXX of the BFM # 0. The lowest bit controls channel 1, followed by the order, the highest bit controls channel 4. Control mode of each character is as follows:

X=0 Preset range -10V~10V (corresponding to -2000~2000)

X=1 Preset range 4mA~20 mA (corresponding to 0~1000)

X=2 Preset range -20mA~20 mA （corresponding to -1000~1000）

X=3 The channel closed

X=4 Preset range -10V~10V (corresponding to -10000~10000)

X=5 Preset range 4mA ~20 mA (corresponding to 0~10000)

X=6 Preset range -20mA ~20 mA （corresponding to -10000~10000）

For example: BFM # 0 is the H1230, says Channel 1 is-10V ~ 10V; channel 2 closed; channel 3 for-20mA ~ 20 mA; channel 4 for the 4mA ~ 20 mA.

If channel is not used, can be turned off, you can not turn off. Turning off the channel do not take conversion time (BFM # 15). In the case channel 2 closed, the entire conversion time is conversion time of three channels not closed (3 × BFM # 15).

BFM #1~#4 the number of average sampling,Sample values corresponding to Each channel (BFM # 9 ~ # 12) accumulate number of samples (BFM # 1 ~ # 4) and then divided by the number of number of samples (BFM # 1 ~ # 4), stored (BFM # 5 ~ # 8)

BFM #5~#8 Store the average sample value

BFM #9~#12 Store Instantaneous sample value

BFM #15 ADC rate time

The time required of each channel is converted once. Note the time required to update the data is the time of BFM # 15 multiplied by the numbers of not closed channels.

For example: BFM # 0 is H3310, BFM # 1 is K7, BFM # 2 is K6, BFM # 15 isK10. The data refresh time of BFM # 9 and BFM # 10 is BFM # 0 × BFM # 15 = 2 × 10 = 20MS. The data refresh time of BFM # 5 is BFM # 0 × BFM # 15 × BFM # 1 = 2 × 10 × 7 = 140MS. The data refresh time of BFM # 6 is BFM # 0 × BFM # 15 × BFM # 2 = 2 × 10 × 6 = 120MS. FROM / TO instruction consumes more time in programs. So the module parameters data of collecting BFM # 5 can use LDP M8012 FROM K0 K5 D10 K1 instruction read, and it's effect is the same as LD M8000 FROM K0 K5 D10 K1. But M8000-driven instruction read once in each scan cycle, greatly extended the program scan cycle.

BFM #20 return to default value

20 is set to 1 will be restored to default values.

BFM #21~#24 The definition and setting method of offset and gain:

Offset and gain can be set independently or together. The normal gain setting range is 1V ~ 15V or 4mA ~ 32mA. The normal offset value is set to the range -5V ~ 5V, or -20mA ~ 20mA.

Before Gain / offset setting, BFM # 21 has to be set to 1 at first, then BFM # 23/24 is modified; then allow offset gain BFM # 22 of each channel is open. When change is completed, BFM # 21 should be set to 2, avoid to be changed again.

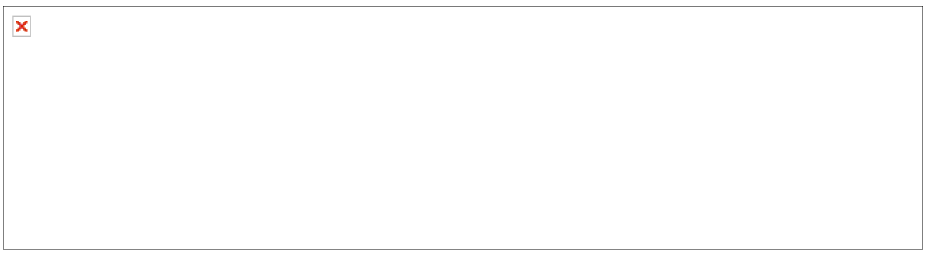
Note the offset gain of the channel needed to modify are the same. You can not set the channel offset data for 1000, while another channel offset data for 1200.

For example: In BFM # 0 is in Mode 0, modify the offset and gain of channel 1, channel 2, respectively 0.5V and 6V. You need to do the following:

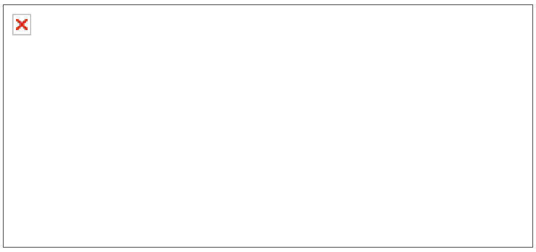
First change BFM # 21 to 1; 300MS later the K500 and K6000 are sent to BFM # 23 and BFM # 24; 300MS later again open to allow the gain BFM # 22. BFM # 22 in this example should be binary 00001111, that is BFM # 22 is amended to H000F; modification is completed. Finally, the BFM # 21 is changed to 2, to prevent further modification.

Program example 1:

A H2U-4AD PLC expansion module connected to the rear of the main module. Number module # 0 according to rule. CH1 port need to collect -10V ~ 10V voltage signal, CH2 need to collect 4 ~ 20mA current signal, CH3/CH4 are not used. Asked to change filter number as 6. The data collected from two channels are stored in D10, D11. User program written as follows:

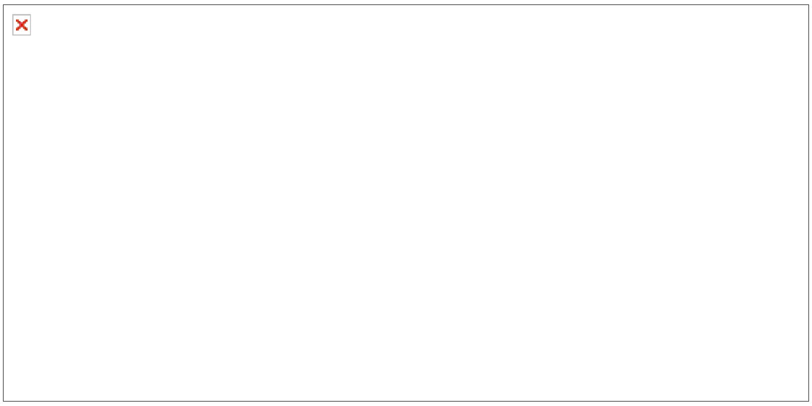


If H2U-4ADR remote module in the example, CAN station is number 1. This example program is as follows:



Program example 2:

For a 4AD module with No # 0 address, called for the adoption of a button X20 to trigger the setting operation of its channel CH1 offset / gain. The program is as follows:



H2U-2AD H2U-2ADR Dual channel analog input module

The signal detection of dual channel analog input can be achieved by 2AD(R) extended module , to convert -10V~10Vor -20mA~20mA to 12bit digital signal applied for PLC main doule. The main module accesses the BFM unit in the extended module register via FROM/TO instruction, achieved read-write parameter.

Parameter specification:

	Instruction

Item			
Conversion speed		15ms/channel(normal speed),6ms/channel (fastspeed),1ms/channel (fastest speed)	
Analog input range	Voltage input	-10~10V DC,(Input impedance is 200KΩ)	The input range could be selected by setting BFM.
	Current input	-20~20mA(Input impedance is 250Ω)	
Digital output		Default setting:–2048 ~ +2047	
Dpi	Voltage input	5mV	
	Current input	10uA	

Module user interface instruction:

Item	Instruction	
The function of connecting termina	V+: Channel voltage positive input\input terminal; I+: Channel current positive input\output terminal; VI-: Channel input\output common terminal; 24+/24-: external 24V input terminal; GND: Protective grounding.	
Indicator	Local extended module	PWR: lighted when it is connected with powered main module. COM: When module runs normally, it flashes in high speed; when there is fault, it flashes in low speed. 24V: When external 24V is connected, it is lighted.
	Remote extended module	PWR: Module 24V power supplies normally COM: It lighting indicates that the module will make communication. ERR: It lighting indicates that there is error.
	Local extended module	The 26-pin ladder connector is applied for the extended input port, and module is connected via flat cable; the 26-pin ladder connector is also

		applied for the extended output port.
Expended port	Remote extended module	The 5-pin hole plug is applied for extended input, and it is recommended to use spiral-eight shield cable or network line as transmission cable.

Address numbering of local special extended module:

The address numbering rule of main module for special modules is: started with the module closed to PLC main module, they are numbered with #0, #1, …,#7 in order, and the IO extended modules are not involved.

Address numbering of remote special extended module:

In remote extended module, the address is: module communication station number+100, and at most 62 remote extended modules are allowed. By toggling the code switch on Station No., the module station number could be set.

If A5, A4=ON, and other bits are all OFF, which indicates the binary address: 011000, decimal number K24, the module number is K24+100 (#124) in FROM/TO instruction programming.

The uniqueness of CAN address should be paid attention to, which means there should not be any same address.

Access BFM area of 2AD(R) module:

The digital AD conversion result is read by PLC main module by reading register buffer unit (BFM) of 2AD(R) module, and the module state is set by modifying specified BFM.PLC main module accesses these BFM units via read/write instruction FROM/TO.

The extended module is equipped with EEPROM storage unit, which is used for saving several BFM setting value, such as signal type, deviant, gain value of every analog input channel, and these units are automatically saved by setting corresponding BFM unit state.

Every register width of BFM area is 16bit (1Word), and according to 2AD (R) BFM area the BFM is defined as following table:

BFM	R/W attribution	Content
#0(E)	WR	Selecting channel signal modes, each HEX bit represents a input channel. In lower 8bit of 2AD (R) module, the higher HEX bit is ch2, and lower HEX bit is ch1: (default value=H00) 0=-10V~10V;the corresponding digital output:-2000~2000 1=4mA~20mA;the corresponding digital

		output:0~1000 2=-20mA~20mA;the corresponding digital output:-1000~1000 3= the channel is closed; 4=-10V~10V;the corresponding digital output:-10000~10000 5=4mA~20mA;the corresponding digital output: 0~10000 6=-20mA~20mA;the corresponding digital output:-10000~10000	
#1	WR	Channel 1	Average filter constant, which is the sampling number used for average calculation with setting range of 1~4096, default value =8. Setting to 1 means high-speed sampling. When BFM#15 changes, it automatically is reset to default value.
#2	WR	Channel 2	
#3~4	-	Reserved	
#5	R	Channel 1	The sampling data of input channel after average filter.
#6	R	Channel 2	
#7~8	-	Reserved	
#9	R	Channel 1	The sampling data of input channel means the instantaneous value before filter processing.
#10	R	Channel 2	
#11~14	-	Reserved	
#15	WR	ADC speed selection	0=normal speed, 15ms/channel （default value）; 1=fast conversion,6ms/channel; 1000~30000=high-speed sampling, which is corresponding to 1ms~30ms/channel

<table>
<tr><td>#16~19</td><td>-</td><td colspan="9">Reserved</td></tr>
<tr><td>#20(E)</td><td>WR</td><td colspan="9">1=reset setting parameter to default value (production value).Default value=0</td></tr>
<tr><td>#21(E)</td><td>WR</td><td colspan="9">2=adjusting deviation/gain is forbidden ;
1=adjusting deviation/gain is allowed (default value)</td></tr>
<tr><td rowspan="2">#22(E)</td><td rowspan="2">WR</td><td>The lower 4bit is corresponding to 2 channels</td><td>-</td><td>-</td><td>-</td><td>-</td><td>G2</td><td>O2</td><td>G1</td><td>O1</td></tr>
<tr><td colspan="9">Enable deviation/gain adjustment. When it is not 0, the module write BFM23/24 value to the internal channel control register.</td></tr>
<tr><td>#23(E)</td><td>WR</td><td colspan="9">Deviation value: when digital output is 0, the initial value of analog input value (0,1,2 mode) is 0</td></tr>
<tr><td>#24(E)</td><td>WR</td><td colspan="9">Gain value: when digital output is +1000, the initial value of analog input value (0,1,2 mode) is 5000</td></tr>
<tr><td>#25~26</td><td>-</td><td colspan="9">Reserved</td></tr>
<tr><td>#27</td><td>R</td><td colspan="9">2AD module software version</td></tr>
<tr><td>#28</td><td>-</td><td colspan="9">Reserved</td></tr>
<tr><td>#29</td><td>R</td><td colspan="9">Error state</td></tr>
<tr><td>#30</td><td>R</td><td colspan="9">Extended module identification code, the ID code of H2U-2AD (R) is K2011</td></tr>
<tr><td>#31</td><td>-</td><td colspan="9">Reserved, can not be accessed</td></tr>
</table>

Where, the definition of state information word BFM #29 is described as following:

BFM#29 bit number	ON state	OFF state
b0:	There is an error.If anyone of b0~b3 is not 0, AD conversion is stopped	No error
b1:	The deviation/gain setting is	The deviation/gain

	error in module EEPROM	data is correct.
b2:	(impossibility)	Power runs normally
b3:	Module hardware faulty	Hardware runs normally
b10:	The value exceeds the rang of -2048~2047	Data output value is normal
b11:	The sampling filter constant exceeds the range of 1~4096	The sampling filter constant is normal
b12:	It is forbidden that BFM#21 value is set to K2	It is allowed that BFM#21 value is set to K2
The other bits bit4~7, bit13~15 in BFM#29 is not defined.		

Where, the BFM unit marked with "(E)" in table is the item saved in EEPROM, which has the characteristic of power failure holding.

The description of register rewriting:

Rewriting BFM #0, #23, #24, which are marked with (E), will cause the module internal writing operation to EEPROM, and writing operation needs a certain time about 300ms per word. So, when rewriting multiple BFM units marked with (E) in PLC programming, after writing a above BFM unit in user program, there should be a delay time instead of continuous writing operation, which ensures that the writing instruction is correctly accomplished.

The description of partial BFM area:

BFM0# selecting channel

Two channels are initialed as -10V~10V in default, controlled by BFM0# hex HXX. The lower bit controls channel 1, and higher bit controls channel 2. The control method of each character are listed as following:

X=0 The pre-set range -10V~10V (corresponding value -2000~2000)

X=1 The pre-set range -4mA~20mA (corresponding value 0~1000)

X=2 The pre-set range -20mA~20mA (corresponding value -1000~1000)

X=3 the channel is closed;

X=4 The pre-set range -10V~10V (corresponding value -10000~10000)

X=5 The pre-set range 4mA~20mA (corresponding value 0~1000)

X=6 The pre-set range -20mA~20mA (corresponding value -10000~10000)

For example: BFM#0=H30 means channel 1 is -10V~10V and channel 2 is closed.

The unused channel can be closed or unclosed, and the closed channel will not occupy conversion time (BFM#15). In example, channel 2 is closed, then the conversion time of the whole channel = channel 1 × BFM#15.

BFM #1~#2 average sampling number

After the sample value in each channel (BFM#9~#10) is added with sampling number (BFM#1~#2), the sum is divided by sampling number (BFM#1~#2), and the result is saved to (BFM #5~#6)

BFM #5~#6 for saving average sampling value

BFM #9~#10 for saving instant sampling value

BFM #15 ADC speed time

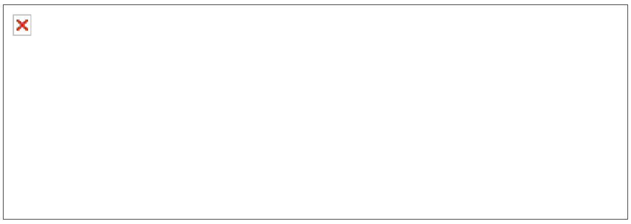
The channel conversion will cost a certain time, and the required time of updating data is BFM#15×the unclosed channel number.

For example: BFM #0=H10，BFM #1=K7，BFM #2=K6，BFM #15=K10; the time of updating BFM #9 and BFM #10 data = BFM#0×BFM#15=2×10=20MS; the time of updating BFM #5 data = BFM#0×BFM#15×BFM#1=2×10×7=140MS; the time of updating BFM #6 = BFM#0×BFM#15×BFM#2=2×10×6=120MS.In program, FROM/TO instruction is time-consuming, so the module parameter can read BFM#5 data in program via LDP M8012? FROM K0 K5 D10 K1 or LD M000?? FROM K0 K5 D10 K1. The latter one applying M8000 instruction will read once in each scanning period, which greatly enlarge the scanning period of the program.

BFM #20 Return to default value

Setting #20 to 1can return to the default value.

BFM #21~#24 the definition and setting method of deviation and gain .



Deviation and gain can be set together or separately, and the setting range of the normal gain value is 1V~15V or 4mA~32mA. The setting range of the normal deviation value is -5V~5V, or -20mA~20mA.

Before setting gain/deviation, MFM#21 should be set to 1 firstly, and then BFM#23/24 is modified; The allowable deviation and gain of each channel BFM#22 is modified, after which BFM#21 should be set to 2 to avoid to be modified again.

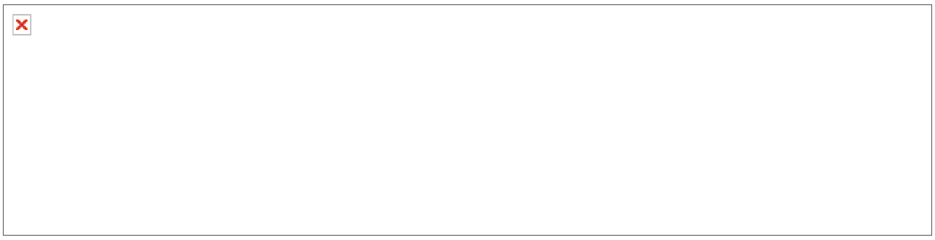
Note: The deviation gain value for each channel should be same. It is forbidden that one channel deviation value is 1000, and the other channel deviation value is 1200.

For example: when in BFM#0=0 mode, it requires that the deviation and gain of channel 1 and channel 2 should be respectively modified to 0.5V and 0.6V, and it should be operated according to the following steps:

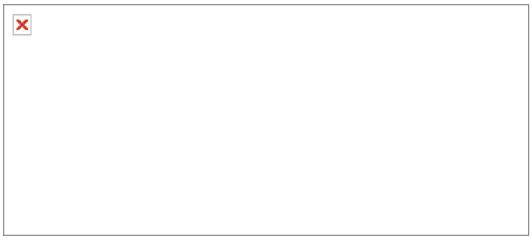
1. Modifying BFM#21 to 1;2. After 300ms, K5000 and K6000 are respectively transmitted to BFM#23 and BFM#24;3. After another 300ms, gain BFM#22 is set. In the example, BFM#22 should be binary 1111(HF);4. End . Finally, BFM#21 should be modified to 2 to avoid to be modified again.

Programming Illustration 1:

A H2U-2AD extended module is connected following PLC main module, which is numbered as #0 module according to numbering rule; CH1 port samples voltage signal of -10V~10V, and CH2 port samples current signal of 4~20mA. The filter number should be modified to six, and the data sampled from two channels are respectively saved to D10, D11. The user program is listed as following:



In example, if it is changed to H2U-2ADR remote module, the CAN station number is 1. The example program is listed as following:



Programming Illustration 2:

For a H2U-2AD module numbered as #0, the deviation/gain setting operation for CH1 channel is activated by X20 button. The program is listed as following:

H2U-4DA H2U-4DAR four channel analog input module

Four analog signal channels output can be achieved by 4DA(R) extended module , and each channel has voltage output and current signal output port with signal amplitude of -10~10V or 0~20mA. The main module accesses the BFM unit in the extended module register via FROM/TO instruction, to achieve controlling analog output signal.

Parameter specification:

Item	Indication	Description
Voltage output signal level	-10V~10V	The voltage signal type could be separately selected for each channel. According to the connected signal type, user can setting corresponding BFM area.
The minimum resistance allowed by voltage chnnel	2kO	
Current output signal	4mA~20mA	
The resistance allowed by current channel	100O~500O	
The number of output channels	Four channels	
Signal conversion rate	4ms/channel	
Conversion speed	15ms/channel (normal speed), 6ms/channel (fastest)	
Digital output	Default setting: -2000 ～ 2000	Allowed range: -10000 ~ 10000
Voltage signal resolution	5mV	Corresponding to 10V/2000
Current signal resolution	20uA	Corresponding to 20mA /1000
Precision	±1% full range	
Occupying I/O points	none	
Isolation design	The photo-coupler should be applied for isolating analog circuit and digital circuit; DC/DC should be applied for isolating analog circuit and external power; The isolation is not needed between analog input signal channels.	

Address numbering of local special extended module:

When PLC is powered,respectively the special modules and IO extended ports are numbered. User cannot intervent or change the numbering result, unless the module connection order is modified.The address numbering rule of main module for special modules is: started with the module closed to PLC main module, they are numbered with #0, #1,…,#7 in order, and the IO extended modules are not involved.

Address numbering of remote special extended module:

In remote extended module, the address is: module communication station number+100, and at most 62 remote extended modules are allowed. By toggling the code switch on Station No., the module station number could be set.

If A5, A4=ON, and other bits are all OFF, which indicates the binary address: 011000, decimal number K24, the module number is K24+100 (#124) in FROM/TO instruction programming.

If code switch is modified, except for matched resistance, baud rate and address will not take effect immediately. After the system is powered again, the new setting parameter will take effect. The uniqueness of CAN address should be paid attention to, which means there should not be any same address.

Access BFM area of 4DA(R) module:

PLC main module writes data value in the register buffer unit (BFM area) of 4DA(R) module, which is converted to analog output by 4DA(R), and module state is set by modifying special BFM area. PLC main module accesses these BFM units via read/write instruction FROM/TO.

The extended module is equipped with EEPROM storage unit, which is used for saving several BFM setting value, such as signal type, deviant, gain value of every analog input channel, and these units are automatically saved by setting corresponding BFM unit state.

Every register width of BFM area is 16bit (1Word), and according to 4DA(R) BFM area the BFM is defined as following table:

BFM	R/W attribution	Content	
#0(E)	WR	Selecting output mode, each HEX bit represents one input channel; the highest bit is ch4, and the lowest bit is ch1: (default value=H0000) 0=-10V~10V; the corresponding digital output:-2000~2000 1=4mA~20mA; the corresponding digital output:0~1000 2=0mA~20mA; the corresponding digital output:0~1000 4=-10V~10V; the corresponding digital output:-10000~10000 5=4mA~20mA; the corresponding digital output:0~10000 6=0mA~20mA; the corresponding digital output:0~10000	
#1	WR	Channel 1	Channel output value, initial value = 0
#2	WR	Channel 2	

#3	WR	Channel 3			
#4	WR	Channel 4			
#5(E)	WR	When PLC shuts down, the data value is held; each HEX bit represents a channel (Hxxxx, the highest bit is CH4, and lowest bit is CH1),when: x=0, the output before shut down is held; x=1, the output is reset to the deviation pre-set value			
#6~7	-	Reserved			
#8(E)	WR	G2	O2	G1	O1
		CH2/CH1 setting deviation/gain command, which is set according to HEX(4 binary bit)bit, initial value =H0000 0=forbid to modify; 1=allow to modify EEPROM data			
#9(E)	WR	G4	O4	G3	O3
		CH3/CH4 setting deviation/gain command, which is set according to HEX(4 binary bit)bit, initial value =H0000 0=forbid to modify; 1=allow to modify BFM data			
#10	WR	Deviation data CH1	Unit: mV or µA Initial deviation value: 0 Initial gain value: +5000, according to mode 0		
#11	WR	Gain data CH1			
#12	WR	Deviation data CH2			
#13	WR	Gain data CH2			
#14	WR	Deviation data CH3			
#15	WR	Gain data CH3			
#16	WR	Deviation data CH4			
#17	WR	Gain data CH4			
#18~19	-	Reserved			
#20(E)	WR	Initial value = 0, when writing with 1, all BFM units are initialled to default value.			

#21(E)	WR	1=allow to adjust output characteristic (initial value); 2= forbid to adjust output characteristic
#22~26	-	Reserved
#27	R	Extended module software version
#28	-	Reserved
#29	R	Error state word
#30	R	Module identity code, 4DA(R) module identity code=K3020
#31	-	Reserved

Where, the definition of state information word BFM #29 is described as following:

BFM#29 bit number	ON state	OFF state
b0:	There is an error. If anyone of b1~b3 is not 0, D/A conversion is stopped	No error
b1:	The deviation/gain setting is error in module EEPROM	The deviation/gain data is correct.
b2:	(impossibility)	Power runs normally
b3:	Module hardware faulty	Hardware runs normally
b10:	Writing data value exceeds the specified range (-2350~2350)	Data output value is normal
b12:	Forbid BFM#21=K1	BFM#21＝K1
The other bits bit4~7, bit11, bit13~15 in BFM#29 is not defined.		

The description of register rewriting:

(E) represents it exits in EEPROM, and it is recommended that the writing operation should not be continually implemented on BFM#8, #9, #20 units, which is with (E) to avoid EEPROM damage.

The modification of BFM#0, which is with (E), will cause module internal writing operation to EEPROM, and writing operation needs about 3s. Note: when in PLC programming, before modifying BFM#10~#17 units, it needs 3s delay time; it is forbidden to implement continuous writing operation and it requires about 300ms delay time before writing operation to ensure that the writing instruction is correctly accomplished.

Note: when the external 24V power supply of the local extended module is power off, the system flag M8067 of PLC main module will be set,error code is D8067=K6708. The flag should be timely checked in programming, the problem can be found in time.

Some explain of BFM area

BFM#0 channel selection

Channel initialization. The 4 default channels are -10V ~ 10V, controlled by hexadecimal HXXXX of the BFM # 0. The lowest bit controls channel 1, followed by the order, the highest bit controls channel 4. Control mode of each character is as follows:

X=0 Preset range -10V~10V （corresponding to -2000~2000）

X=1 Preset range 4mA~20 mA （corresponding to 0~1000）

X=2 Preset range 0mA~20 mA （corresponding to 0~1000）

X=4 Preset range -10V~10V （corresponding to -10000~10000）

X=5 Preset range 4mA ~20 mA （corresponding to 0~10000）

X=6 Preset range 0mA ~20 mA （corresponding to 0~10000）

For example: BFM # 0 is the H1220, says Channel 1 is-10V ~ 10V; channel 2 and channel 3 for 0mA ~ 20 mA; channel 4 for the 4mA ~ 20 mA.

BFM #1~#4 channel output

With TO instruction to write data to the BFM # 1 ~ # 4, you can control the analog output. The initial values are 0.

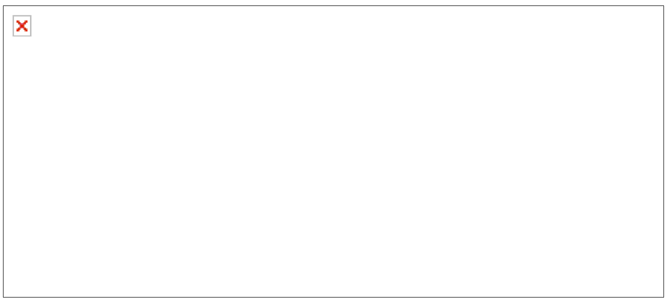
BFM #5 Data retaining mode

When the main module is in the state RUN to STOP, the final model are kept (X = 0) or the offset value (X = 1) output.

For example: BFM # 0 is the H0000, BFM # 5 is the H0101, offset of 4 channels are 0.1V. Changes from the RUN to STOP, BFM # 1 ~ # 4 inside the values are 1500 (7.5V), when STOP, the channel 1 and channel 3 output voltage becomes 0.1V, Channel 2 and Channel 4, the output voltage remains at 7.5V.

BFM # 8 ~ # 17, BFM # 21 The definition of offset and gain setting method:

H2U-4DA (R) There are three operating modes, characteristic curve as shown below:



Among them, Gain value is the analog input value when digital output is 1000; Offset value is the analog input value when digital output is 0. Offset and gain settings can be set independently or together, set the parameters of the unit is mV (mode 0) and μA (mode 1 and 2)

BFM # 8 ~ # 9 are the offset, gain setting instruction, each hex HEX (binary four bits) to control the prohibited or permitted, pay attention to in the AD input module it is each bit in a binary bit to control. The offset, gain setting instruction of DA module and the AD module differ. BFM # 10 ~ # 17 are the offset, gain settings, BFM # 21 are the instruction to set curve characteristics.

Before Gain / offset setting, BFM # 21 has to be set to 1 at first, then relative units of BFM＃10～＃17 are modified; then write the word of allowing operation to BFM#8, #9. When change is completed, BFM # 21 should be set to 2, avoid to be changed again.

For example: In BFM # 0 as a model H0000, the offset and gain of channel 1 needed to modify are 0.2V and 5.5V; offset and gain of channel 3 are 0.5V and 6V. You need to do the following:

First change BFM # 21 to 1; 300MS later the K200、 K5500、K500 and K6000 are sent to BFM#10、#11、#14、#15; Followed by BFM # 8 ~ # 9 to open to allow offset, gain, that is to amend BFM # 8, BFM # 9 as H0011; modification is completed. Finally, the BFM # 21 is changed to 2, to prevent further modification.

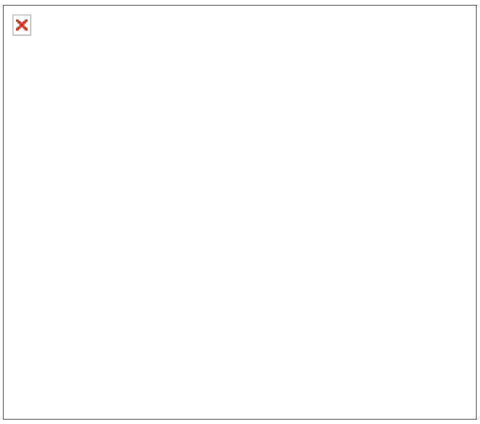
BFM #20 return to default value

The BFM # 20 is set to 1 can be restored to the default values

Program example 1:

A H2U-4AD PLC expansion module connected to the rear of the main module. And next to connect a H2U-4DA expansion module. Number H2U-4DA module #1 according to rule. CH1 port need to output 6mA~20mA current signal, CH2 need to output -10V~10V voltage signal, CH3/CH4 are not used. Ask to close port X21 to trigger 4DA module initialization. User program written as follows:

If H2U-4ADR remote module in the example, CAN station is number 2. This example program is as follows:



Programming example 2:

In the example of local module, if it requires that D100, D101 is outputted from CH1, CH2, and the operation state of 4DA module will be checked every 1s. The program is listed as following:

Where, if the external 24V power supply of 4DA module is off, the system flag M8067 will be set,error code D8067=K6708.

H2U-2DA H2U-2DAR Dual channel analog output module

Dual analog signal channels output can be achieved by 2DA(R) extended module , and each channel has voltage output and current signal output port with signal amplitude of -10~10V or 0~20mA. The main module accesses the BFM unit in the extended module register via FROM/TO instruction, to achieve controlling analog output signal.

Parameter specification

Item		Indication
Conversion speed		2 channels 2.1ms
Analog output range	Voltage output	-10~10V DC(External load resistence is 2KΩ~1MΩ)
	Current output	0~20mA(external load resistence is 500Ω or less)
Digital input		Default setting: -2000~2000, allowable range: -10000~10000
dpi	Voltage output	5mV(10V/2000)
	Current output	10µA(20mA /2000)
Total precision		±1%(corresponding to full range of 10V) ±1%(corresponding to full range of 20mA)

The table of module user interface:

Item	Description
The function of connecting	V+: Channel voltage positive output terminal; I+: Channel current positive output terminal; VI-: Channel output common terminal;

terminal	24+/24-: external 24V input terminal; GND: Protective grounding.	
Indicator	Local extended module	PWR: lighted when it is connected with powered main module. COM: When module runs normally, it flashes in high speed; when there is fault, it flashes in low speed. 24V: When external 24V is connected, it is lighted.
	Remote extended module	PWR: Module 24V power supplies normally COM: It lighting indicates that the module will make communication. ERR: It lighting indicates that there is error.
Expended port	Local extended module	The 26-pin ladder connector is applied for the extended input port, and module is connected via flat cable; the 26-pin ladder connector is also applied for the extended output port.
	Remote extended module	The 5-pin hole plug is applied for extended input, and it is recommended to use spiral-eight shield cable or network line as transmission cable.

Address numbering of local special extended module:

When PLC is powered, all connected extended modules will be checked for one time, and respectively the special modules and IO extended ports are numbered. User cannot intervent or change the numbering result, unless the module connection order is modified.

The address numbering rule of main module for special modules is: started with the module closed to PLC main module, they are numbered with #0, #1,…,#7 in order, and the IO extended modules are not involved.

Address numbering of remote special extended module:

In remote extended module, the address is: module communication station number+100, and at most 63 remote extended modules are allowed. By toggling the code switch on Station No., the module station number could be set.

If A5, A4 in certain module are all ON, and the other bits are OFF, (binary: 011000; decimal K24), the module numbering in FROM/TO instruction is K24+100, (#124)

If code switch is modified, except for matched resistance, baud rate and address will not take effect immediately. After the system is powered again, the new setting parameter will take effect. The uniqueness of CAN address should be paid attention to, which means there should not be any same address.

Access BFM area of 2DA(R) module:

PLC main module writes data value in the register buffer unit (BFM area) of 2DA(R) module, which is converted to analog output by 2DA(R), and module state is set by modifying special BFM area. PLC main module accesses these BFM units via read/write instruction FROM/TO.

The extended module is equipped with EEPROM storage unit, which is used for saving several BFM setting value, such as signal type, deviant, gain value of every analog input channel, and these units are automatically saved by setting corresponding BFM unit state.

Every register width of BFM area is 16bit (1Word), and according to 2DA(R) BFM area the BFM is defined as following table:

<table>
<tr><th>BFM</th><th>R/W attribution</th><th colspan="4">Content</th></tr>
<tr><td>#0(E)</td><td>WR</td><td colspan="4">Selecting output modes, each HEX bit represents an output channel. In lower 8bit of 2DA(R) module, the higher HEX bit is ch2, and lower HEX bit is ch1: (default value=H00)

0=-10V~10V; the corresponding digital output:-2000~2000

1=4mA~20mA; the corresponding digital output:0~1000

2=0mA~20mA; the corresponding digital output:0~1000

4=-10V~10V; the corresponding digital output:-10000~10000

5=4mA~20mA; the corresponding digital output:0~10000

6=0mA~20mA; the corresponding digital output:0~10000</td></tr>
<tr><td>#1</td><td>WR</td><td>Channel 1</td><td colspan="3" rowspan="2">Channel output value, initial value = 0</td></tr>
<tr><td>#2</td><td>WR</td><td>Channel 2</td></tr>
<tr><td>#3~4</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td>#5(E)</td><td>WR</td><td colspan="4">Selecting value holding modes when PLC shuts down, each HEX bit represents an output channel. In lower 8bit of 2DA(R) module, the higher HEX bit is ch2, and lower HEX bit is ch1, when:
x=0, the output before shut down is held; x=1, the output is reset to the deviation pre-set value</td></tr>
<tr><td>#6~7</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td rowspan="2">#8(E)</td><td rowspan="2">WR</td><td>G2</td><td>O2</td><td>G1</td><td>O1</td></tr>
<tr><td colspan="4">CH2/CH1 setting deviation/gain command, which is set according to HEX bit, initial value =H0000
0=forbid to modify; 1=allow to modify EEPROM data</td></tr>
</table>

#9	-	Reserved	
#10	WR	Offset data CH1	Unit: mV or μA Initial deviation value: 0 Initial gain value: +5000, according to mode 0 (0,1,2 modes)
#11	WR	Gain data CH1	
#12	WR	Offset data CH2	
#13	WR	Gain data CH2	
#14~19	-	Reserved	
#20(E)	WR	Initial value = 0, when writing with 1, all BFM units are initialled to default value.	
#21(E)	WR	1=allow to adjust output characteristic (initial value); 2= forbid to adjust output characteristic	
#22~26	-	Reserved	
#27	R	Extended module software version	
#28	-	Reserved	
#29	R	Error state word	
#30	R	Module identity code, 2DA(R) module identity code=K3021	
#31	-	Reserved	

Where, the definition of state information word BFM #29 is described as following:

BFM#29 bit number	ON state	OFF state
b0:	There is an error. If anyone of b1~b3 is not 0, D/A conversion is stopped	No error
b1:	The deviation/gain setting is error in module EEPROM	The deviation/gain data is correct.
b2:	(impossibility)	Power runs normally
b3:	Module hardware faulty	Hardware runs normally
b10:	Writing data value exceeds the specified range	Data output value is normal
b12:	Forbid BFM#21=K1	BFM#21=K1

The other bits bit4~7, bit11, bit13~15 in BFM#29 is not defined.

The description of register rewriting:

(E) represents it exits in EEPROM, and it is recommended that the writing operation should not be continually implemented on BFM#8, #20 units, which is with (E) to avoid EEPROM damage.

The modification of BFM#0, which is with (E), will cause module internal writing operation to EEPROM, and writing operation needs about 3s. Note: when in PLC programming, before modifying BFM#10~#13 units, it needs 3s delay time; it is forbidden to implement continuous writing operation and it requires about 300ms delay time before writing operation to ensure that the writing instruction is correctly accomplished.

Note: when the external 24V power supply of the local extended module is power off, the system flag M8067 of PLC main module will be set,error code D8067=k6708. The flag should be timely checked in programming, the problem can be found in time.

The description of partial BFM area

BFM#0 selecting channel

Two channels are initialed as -10V~10V in default, controlled by BFM0# hex HXX. The lower bit controls channel 1, and higher bit controls channel 2. The control method of each character are listed as following:

X=0 The pre-set range -10V~10V (corresponding value -2000~2000)

X=1 The pre-set range -4mA~20mA (corresponding value 0~1000)

X=2 The pre-set range 0mA~20mA (corresponding value 0~1000)

X=4 The pre-set range -10V~10V (corresponding value -10000~10000)

X=5 The pre-set range 4mA~20mA (corresponding value 0~1000)

X=6 The pre-set range 0mA~20mA (corresponding value 0~10000)

For example: BFM#0=H10 means channel 1 is -10V~10V and channel 2 is 4mA~20mA.

BFM #1~#2 channel output value

Writing data to BFM #1～#2 with TO instruction to control analog output.The initial values are all 0.

BFM #5 Data holding mode

When main module is in the state from RUN to STOP, the final mode of RUN will be held (X=0) or deviation (X=1) output,

For example: BFM#0=H00, BFM#5=H01, the deviation value in two channels are all 0.1V. When it changes from RUN to STOP, the values in BFM#1~#2 are all 1500 (7.5V); After STOP, output voltage in channel 1 changes to 0.1V, and output voltage in channel 2 holds to 7.5V.

BFM #8?BFM #10~#13?BFM #21 the definition and setting method of deviation and gain：

There are totally three operation modes for H2U-2DA(R), and the characteristic curve is shown as following figure:

Where, the gain value is the analog output corresponding to data 1000; and deviation value is the analog output corresponding to data 0. Deviation and gain can be set separately or together, and the parameter unit is mV(mode 0) and μA (mode 1 and 2)

BFM#8 is deviation and gain setting command, and each HEX (which is composed of four binary bits) bit will control forbid or enable. Note: it is every binary bit to control in AD input module, and the deviation and gain setting command are different in DA module and AD module. BFM#10~#13 is deviation, gain setting value, and BFM#21 is curve characteristic setting command.

Before setting gain/deviation, BFM#21 should be set to 1, and then BFM #10~#13 unit values are modified according to requirements; after that, the operation allowable word is written to BFM#8 unit. After modification is accomplished, BFM#21 is set to 2 to avoid to be changed again.

For example: when in BFM#0=H00 mode, it requires that the deviation and gain of channel 1 should be modified to 0.2V and 5.5V, and it should be operated according to the following steps:

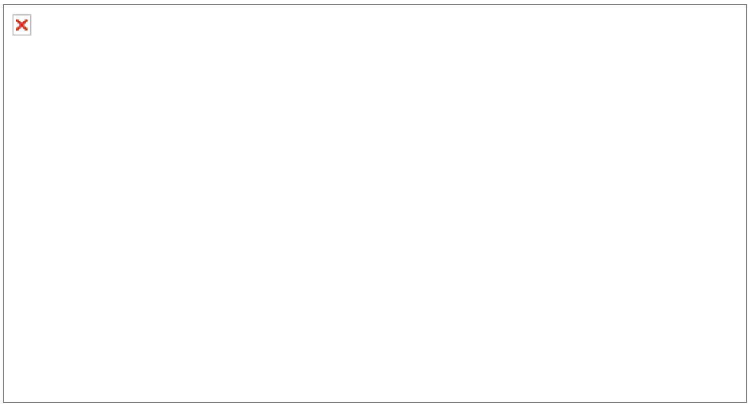
1. Modifying BFM#21 to 1;
2. After 300ms, writing K200, K5500 to BFM#10 #11 respectively;
3. Setting BFM#8 to enable deviation, gain, which means modifying BFM#8 to H0011;
4. Finally, BFM#21 should be modified to 2 to avoid to be modified again.

BFM #20 Return to default value

Setting BFM#20 to 1can return to the default value.

Programming example 1:

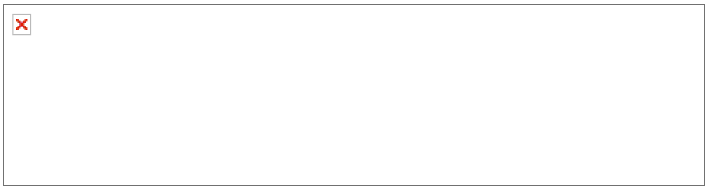
A H2U-2AD extended module is connected to PLC main module, and followed by a H2U-2DA extended module. According to numbering rule, H2U-2DA is numbered as #1 module, in which 6mA~20mA current signal is requested in CH1 port output and -10V~10V voltage signal is requested in CH2 port output. It requests X21 port close will cause initial operation of 2DA module. The user program is listed as following:



In example, if it is changed to H2U-2DAR remote module, the CAN station number is 10. The example program is listed as following:

Programming example 2;

In the example of local module, if it requires that D100, D101 is outputted from CH1, CH2, and the operation state of 2DA module will be checked every 1s. The program is listed as following:



Where, if the external 24V power supply of 2DA module is off, the system flag M8067 will be set,error code D8067=K6708.

H2U-4AM H2U-4AMR Dual channel analog input/ ouput module

The signal detection of dual channel analog input and output of dual channel analog signal can be achieved by H2U local extended module . The signal of 0~10V or 0~20mA can be converted to 12bit digital signal by each input channel for PLC main module reading; each output channel has output port of voltage signal and current signal with signal amplitude of -10~10V or 0~20mA. The main module accesses the BFM unit in the extended module register via FROM/TO instruction, to achieve controlling analog output signal.

Parameter specification:

Item		Indicator in AD section	
Conversion speed		15ms/channel(normal speed),6ms/channel(fast speed),1ms/channel(fastest speed)	
Analog input range	Voltage input	0~10V DC,(Input impedance is 200KΩ)	The input range could be selected by setting BFM.
	Current input	0~20mA(Input impedance is 250Ω)	
Digital output		Default setting: 0 ~ +2000	
dpi	Voltage input	5mV	
	Current input	10uA	
Precision		±1%	
Item		Indicator in DA section	
Conversion speed		2 channels 2.1ms	
Analog output range	Voltage output	-10~10V DC(External load resistence is 2KΩ~1MΩ)	
	Current output	0~20mA(external load resistence is 500Ω or less)	
Digital input		Default setting: -2000~2000, allowable range: -10000~10000	
dpi	Voltage output	5mV(10V/2000)	
	Current output	10μA(20mA /2000)	
Total precision		±1%(corresponding to full range of 10V) ±1%(corresponding to full range of 20mA)	

The table of module user interface:

Item	Description	
The function of connecting terminal	V+:Channel voltage positive intpu\input terminal; I+:Channel current positive input\output terminal; VI-: Channel input\output common terminal; 24+/24-: external 24V input terminal; GND: Protective grounding.	
	Local extended	PWR: lighted when it is connected with powered?

Indicator	module	main module. COM: When module runs normally, it flashes in high speed; when there is fault, it flashes in low speed. 24V: When external 24V is connected, it is lighted.
	Remote extended module	PWR: Module 24V power supplies normally COM: It lighting indicates that the module will make communication. ERR: It lighting indicates that there is error.
Expended port	Local extended module	The 26-pin ladder connector is applied for the extended input port, and module is connected via flat cable; the 26-pin ladder connector is also applied for the extended output port.
	Remote extended module	The 5-pin hole plug is applied for extended input, and it is recommended to use spiral-eight shield cable or network line as transmission cable.

Address numbering of local special extended module:

When PLC is powered, all connected extended modules will be checked for one time, and respectively the special modules and IO extended ports are numbered. User cannot intervent or change the numbering result, unless the module connection order is modified.

The address numbering rule of main module for special modules is: started with the module closed to PLC main module, they are numbered with #0, #1,...,#7 in order, and the IO extended modules are not involved.

Address numbering of remote special extended module:

In remote extended module, the address is: module communication station number+100, and at most 63 remote extended modules are allowed. By toggling the code switch on Station No., the module station number could be set.

If A5, A4 in certain module are all ON, and the other bits are OFF, (binary: 011000; decimal K24), the module numbering in FROM/TO instruction is K24+100, (#124)

If code switch is modified, except for matched resistance, baud rate and address will not take effect immediately. After the system is powered again, the new setting parameter will take effect. The uniqueness of CAN address should be paid attention to, which means there should not be any same address.

Access BFM area of 4AM(R) module:

PLC main module accesses these BFM units via read/write instruction FROM/TO.

The extended module is equipped with EEPROM storage unit, which is used for saving several BFM setting value, such as signal type, deviant, gain value of every analog input /output channel, and these units are automatically saved by setting corresponding BFM unit state.

Every register width of BFM area is 16bit (1Word), and according to 4AM module the BFM area is defined as following table:

<table>
<tr><td>BFM</td><td>R/W attribution</td><td colspan="2">Input channel content</td></tr>
<tr><td>#0(E)</td><td>WR</td><td colspan="2">Selecting channel signal modes, each HEX bit represents a input channel. In lower 8bit of 4AM(R) module, the higher HEX bit is ch2, and lower HEX bit is ch1: (default value=H00)
0=0V~10V; the corresponding digital output:0~2000
1=4mA~20mA; the corresponding digital output:0~1000
2=0mA~20mA; the corresponding digital output:0~1000
3= the channel is closed;
4=0V~10V; the corresponding digital output: 0~10000
5=4mA~20mA; the corresponding digital output:0~1000
6=0mA~20mA; the corresponding digital output:0~1000</td></tr>
<tr><td>#1</td><td>WR</td><td>Channel 1</td><td rowspan="2">Average filter constant, which is the sampling number used for average calculation with setting range of 1~4096, default value =8. Setting to 1 means high-speed sampling. When BFM#15 changes, it automatically</td></tr>
<tr><td>#2</td><td>WR</td><td>Channel 2</td></tr>
</table>

			resumes to default value.
#3~4	-	Reserved	
#5	R	Channel 1	The sampling data of input channel after average filter
#6	R	Channel 2	
#7~8	-	Reserved	
#9	R	Channel 1	The sampling data of input channel means the instantaneous value before filter processing.
#10	R	Channel 2	
#11~14	-	Reserved	
#15	WR	ADC speed selection	0=normal speed, 15ms/channel£¨default value£©; 1=fast conversion,6ms/channel; 1000~30000=high-speed sampling, which is corresponding to 1ms~30ms/channel
#16~19	-	Reserved	
#20(E)	WR	1=reset setting, reset all input channel parameters (BFM#0~BFM#32) to default value.Default value=0	
#21(E)	WR	2=adjusting deviation/gain is forbidden; 1=adjusting deviation/gain is allowed, (default value)	
#22(E)	WR	The lower 4bit is corresponding to 2 channel	- | - | - | - | G2 | O2 | G1 | O1
		Enable offset/gain adjustment. When it is not 0, the module write BFM23/24 value to the internal channel control register.	

#23(E)	WR	Offset value: when digital output is 0, the analog input value (0,1,2 mode)
#24(E)	WR	Gain value: when digital output is +1000, the analog input value (0,1,2 mode)
#25~26	-	Reserved
#27	R	4AM module software version
#28	-	Reserved
#29	R	Input channel error state
#30	R	Extended module identification code, the ID code of H2U-4AM (R) is K4051
#31	-	Reserved,cannot be accessed

BFM	R/W attribution	Output channel content	
#32(E)	WR	Selecting output modes, each HEX bit represents an output channel. In lower 8bit of 2DA(R) module, the higher HEX bit is ch2, and lower HEX bit is ch1: (default value=H00) 0=-10V~10V; the corresponding digital output:-2000~2000 1=4mA~20mA; the corresponding digital output:0~1000 2=0mA~20mA; the corresponding digital output:0~1000 4=-10V~10V; the corresponding digital output:-10000~10000 5=4mA~20mA; the corresponding digital output:0~1000 6=0mA~20mA; the corresponding digital output:0~10000	
#33	WR	Channel 3	Channel output value, initial value = 0

<table>
<tr><td>#34</td><td>WR</td><td>Channel 4</td><td colspan="3"></td></tr>
<tr><td>#35~36</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td>±37(E)</td><td>WR</td><td colspan="4">Selecting value holding modes when PLC shuts down, each HEX bit represents an output channel. In lower 8bit of 4AM(R) module, the higher HEX bit is ch4, and lower HEX bit is ch3, when:

x=0, the output before shut down is held;
x=1, the output is reset to the deviation pre-set value</td></tr>
<tr><td>#38~39</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td rowspan="2">#40(E)</td><td rowspan="2">WR</td><td>G2</td><td>O2</td><td>G1</td><td>O1</td></tr>
<tr><td colspan="4">CH4/CH3 setting deviation/gain command, which is set according to HEX bit, initial value =H0000

0=forbid to modify; 1=allow to modify EEPROM data</td></tr>
<tr><td>#41</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td>#42</td><td>WR</td><td>Offset data CH3</td><td colspan="3" rowspan="4">Unit: mV or μA

Initial deviation value: 0

Initial gain value: +5000, according to mode 0(0,1,2 modes)</td></tr>
<tr><td>#43</td><td>WR</td><td>Gain data CH3</td></tr>
<tr><td>#44</td><td>WR</td><td>Offset data CH4</td></tr>
<tr><td>#45</td><td>WR</td><td>Gain data CH4</td></tr>
<tr><td>#46~51</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td>#52(E)</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td>#53(E)</td><td>WR</td><td colspan="4">1=allow to adjust output characteristic (initial value); 2= forbid to adjust output characteristic</td></tr>
<tr><td>££54~60</td><td>-</td><td colspan="4">Reserved</td></tr>
<tr><td>££61</td><td>R</td><td colspan="4">Output channel error state</td></tr>
</table>

Where, the definition of state information word BFM #29 is described as following:

BFM#29 bit number	ON state	OFF state
b0:	There is an error.If anyone of b0~b3 is not 0, AD conversion is stopped	No error
b1:	The offset/gain setting is error in module EEPROM	The offset/gain data is correct.
b2:	(impossibility)	Power runs normally
b3:	Module hardware faulty	Hardware runs normally
b10:	The value exceeds the rang of -2048~2047	Data output value is normal
b11:	The sampling filter constant exceeds the range of 1~4096	The sampling filter constant is normal
b12:	It is forbidden that BFM#21 value is set to K2	It is allowed that BFM#21 value is set to K2
The other bits bit4~7, bit13~15 in BFM#29 is not defined.		

Where, the definition of state information word BFM #61 is described as following:

BFM £ £61 bit number	ON state	OFF state
b0:	There is an error.If anyone of b1~b3 is not 0, D/ A conversion is stopped.	No error
b1:	The offset/gain setting is error in module EEPROM	The offset/gain data is correct.
b2:	(impossibility)	Power runs normally
b3:	Module hardware faulty	Hardware runs normally
b10:	The input value exceeds the rang.	Data output value is normal
b12:	It is forbidden that BFM#21 value is set to K1	BFM#21=K1
The other bits bit4~7, bit13~15 in BFM#61 is not defined.		

Where, the BFM unit marked with "(E)" in table is the item saved in EEPROM, which has the characteristic of power failure holding.

The description of register rewriting:

Rewriting BFM #0, #32, which are marked with (E), will cause the module internal writing operation to EEPROM, and writing operation needs a certain time about 300ms per word. So, when rewriting multiple BFM units in PLC programming, after writing a above BFM unit in user program, there should be a delay time instead of continuous writing operation, which ensures that the writing instruction is correctly accomplished.

Note: when the external 24V power supply of the local extended module is power off, the system flag M8067 of PLC main module will be set,error code D8067=K6708.The flag should be timely checked in programming, the problem can be found in time.

Explanation of input channel BFM section:

BFM0# selecting input channel

Two input channels are initialled as 0~10V in default, controlled by BFM0# hex HXX. The lower bit controls channel 1, and higher bit controls channel 2. The control method of each character is listed as following:

X=0 The pre-set range 0V~10V (corresponding value 0~2000)

X=1 The pre-set range 4mA~20 mA (corresponding value 0~1000)

X=2 The pre-set range 0mA~20 mA (corresponding value 0~1000)

X=3 The channel is closed;

X=4 The pre-set range 0V~10V (corresponding value 0~10000)

X=5 The pre-set range 4mA ~20 mA (corresponding value 0~10000)

X=6 The pre-set range 0mA ~20 mA (corresponding value 0~10000)

For example: BFM#0=H30 means channel 1 is 0V~10V and channel 2 is closed.

The unused channel can be closed or unclosed, and the closed channel will not occupy conversion time (BFM#15). In example, channel 2 is closed, then the conversion time of the whole channel = channel 1 × BFM#15.

BFM #1~#2 average input sampling number

After the sample value in each channel (BFM#9~#10) is added with sampling number (BFM#1~#2), the sum is divided by sampling number (BFM#1~#2), and the result is saved to (BFM #5~#6)

BFM #5~#6 for saving average input sampling value

BFM #9~#10 for saving instant input sampling value

BFM #15 input ADC speed time

The input channel conversion will cost a certain time, and the required time of updating data is BFM#15¡Áthe unclosed channel number.

For example: BFM #0=H10,BFM #1=K7,BFM #2=K6,BFM #15=K10; the time of updating BFM #9 and BFM #10 data = BFM#0×BFM#15=2×10=20MS; the time of updating BFM #5 data = BFM#0×BFM#15×BFM#1=2×10×7=140MS; the time of updating BFM #6 = BFM#0×BFM#15×BFM#2=2×10×6=120MS. In program, FROM/TO instruction is time-consuming, so the module parameter can read BFM#5 data in program via LDP M8012 FROM K0 K5 D10 K1 or LD M000 FROM K0 K5 D10 K1. The latter one applying M8000 instruction will read once in each scanning period, which greatly enlarge the scanning period of the program.

BFM #20 resetting input channel to default value

Setting #20 to 1can return to the default value

BFM #21~#24 the definition and setting method of input channel offset and gain:

There are totally two operation modes for H2U-4AM(R) input channel, and the characteristic curve is shown as following figure:

Deviation and gain can be set together or separately, and the setting range of the normal gain value is 1V~15Vor 4mA~32mA. The setting range of the normal deviation value is 0V~5V, or 0mA~20mA.

Before setting gain/deviation, MFM#21 should be set to 1 firstly, and then BFM#23/24 is modified; The allowable deviation and gain of each channel BFM#22 is modified, after which BFM#21 should be set to 2 to avoid to be modified again.

Note: The deviation gain value for each channel should be same. It is forbidden that one channel deviation value is 1000, and the other channel deviation value is 1200.

For example: when in BFM#0=0 mode, it requires that the deviation and gain of channel 1 and channel 2 should be respectively modified to 0.5V and 0.6V, and it should be operated according to the following steps:

1. Modifying BFM#21 to 1;2. After 300ms, K5000 and K6000 are respectively transmitted to BFM#23 and BFM#24;3. After another 300ms, gain BFM#22 is set. In the example, BFM#22 should be binary 1111(HF);4.Finally, BFM#21 should be modified to 2 to avoid to be modified again.

Explanation of output channel BFM section

BFM#32 output channel selection

Two output channels are initialled as -10V~10V in default, controlled by BFM32# hex HXX. The lower bit controls channel 3, and higher bit controls channel 4. The control method of each character is listed as following:

X=0 The pre-set range -10V~10V (corresponding value -2000~2000)

X=1 The pre-set range 4mA~20 mA (corresponding value 0~1000)

X=2 The pre-set range 0mA~20 mA (corresponding value 0~1000)

X=4 The pre-set range -10V~10V (corresponding value -10000~10000)

X=5 The pre-set range 4mA ~20 mA (corresponding value 0~10000)

X=6 The pre-set range 0mA ~20 mA (corresponding value 0~10000)

For example: BFM#32=H10 means channel 3 is -10V~10V and channel 4 is 4mA~20mA.

BFM #33~#34 output channel output value

Writing data to BFM #33~#34 with TO instruction to control analog output.The initial values are all 0.

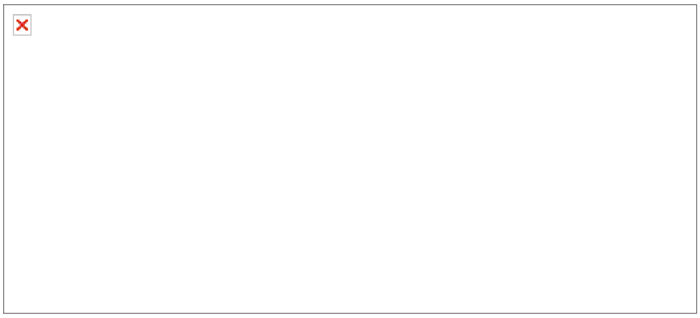
BFM #37 output channel data holding mode

When main module is in the state from RUN to STOP, the final mode of RUN will be held (X=0) or deviation (X=1) output.

For example: BFM#32=H00, BFM#37=H01, the deviation value in two channels are all 0.1V. When it changes from RUN to STOP, the values in BFM#33~#34 are all 1500 (7.5V); After STOP, output voltage in channel 4 changes to 0.1V, and output voltage in channel 2 holds to 7.5V.

BFM #40,BFM #42~#45,BFM #53 the definition and setting method of output channel deviation and gain:

There are totally three operation modes for H2U-4AM(R) output channel, and the characteristic curve is shown as following figure:



The gain is the corresponding analog output when the numerical value is 1000, while the offset is the corresponding output when the numerical value is 0. The offset and gain can be configured individually or collectively. The unit for setting parameters is mV (mode 0) and ¦ÌA (mode 1 and 2)

BFM#40 is the setting command for offset and gain values. Every HEX bit (composed of 4 binary bit) in the hexadecimal prohibits or permits the command. Please note that the AD input module is controlled by every binary bit, while the DA and AD modules possesses differentiation in gain and offset setting commands respectively. BFM #10~#13 are the offset/gain settings, and BFM #53 is the curve characteristic setting command.

Before setting gain/offset, MFM#53 should be set to 1 firstly, and then the value correlate to BFM#42~#45 is modified by needed; The allowable operation word be wrote to BFM#40 , after which BFM#53 should be set to 2 to avoid to be modified again.

For example, when BFM#40 is under the H00 mode, the offset and gain values that require channel modifications are 0.2V and 5.5V. The operation procedure is as follows:

First, modify BFM#53 to 1; after passes 300MS, transfer K200 and K5500 to BFM#42 and #43 respectively; immediately switch BFM#40 on to allow offset/gain command to modify BFM#40 to H0011. The modification is now complete. Modify BFM#53 to 2 to prevent other changes.

BFM #52 Output channel returns to default factory settings

Modify BFM#52 to 1 to return to the default values.

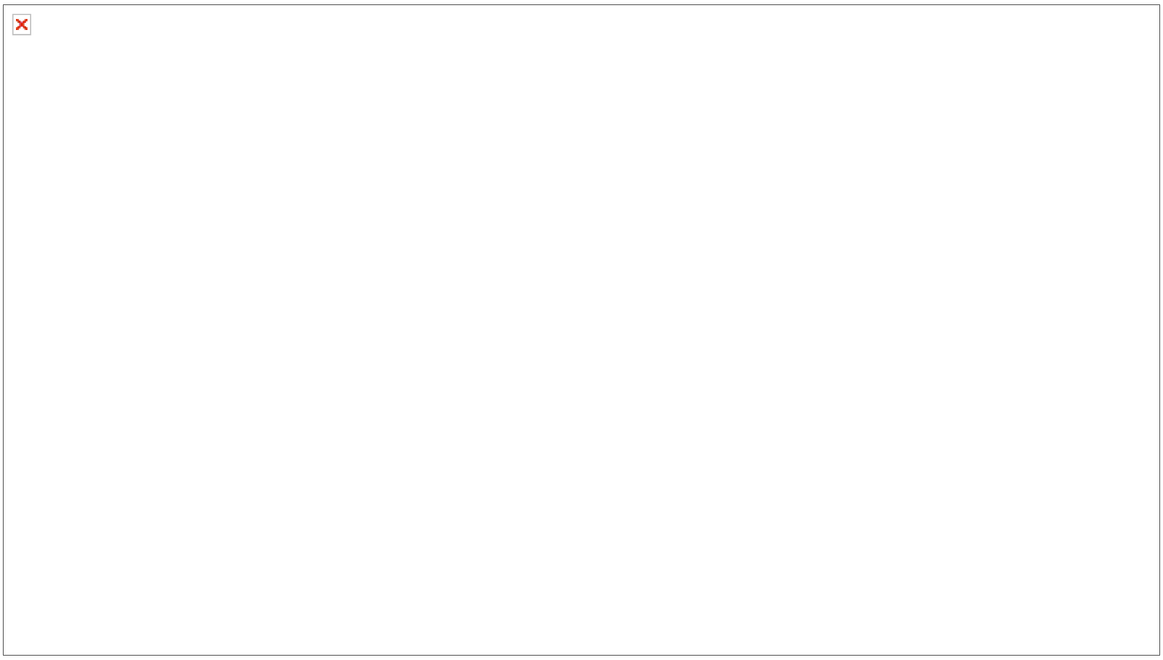
Programming Example 1:

The 4AM module 2 input channels(ch1,ch2)input port connect to the 4~20mA signal,corresponding gathered data range is 0~1000;set 2 output channels in it to output 0~10V signal, corresponding data is 0~2000.

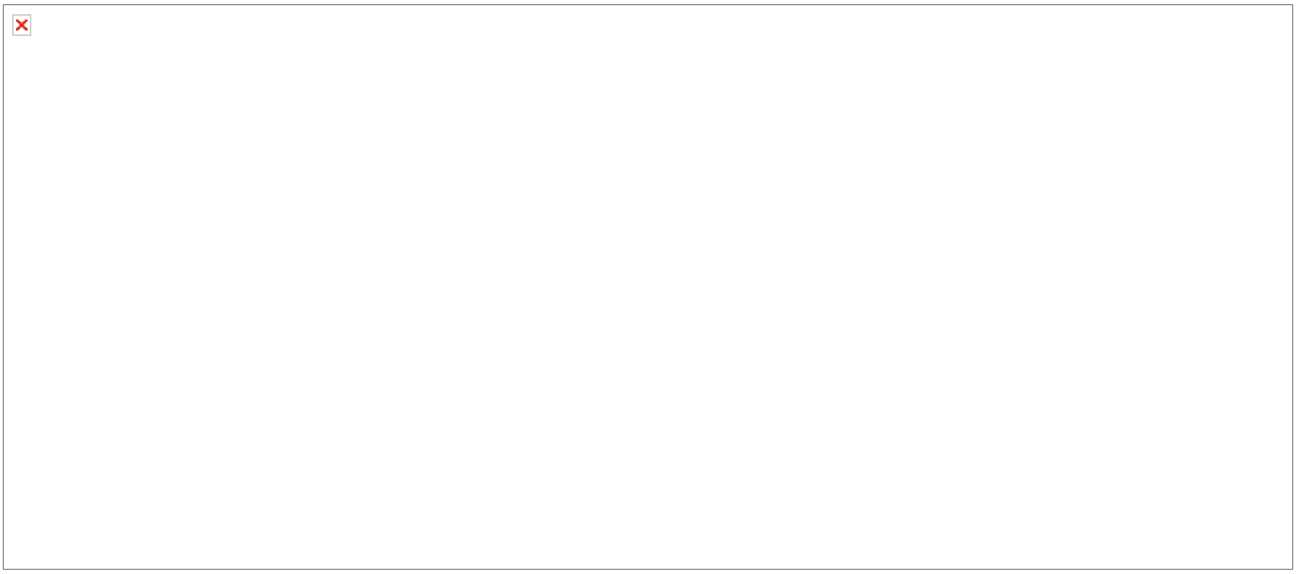
According to definition that ch1 and ch2 set to BFM#0,BFM#0=H0011

According to definition that ch3 and ch4 set to BFM#32,BFM#32=H0000

so can do the program:



The readed ADC data need linearity calculation,then reading can be same as actual materiel.commonly in range of materiel quantity,measure two reading separately,make use of linear interpolation mode,gained the correction algorithm,sentence as follows,use DMUL,DDIV,aviod overflow of calculation,for example:



According to data read from ch1(D101) of 4AM, count out actual reading D299 be measured .

Programming Example 2:

In one H2U-4AMR remote expansion module, the CAN station number is 12. In the module, the CH1 port requires 0~10V of signals to modify filter frequency to 6 and save the collected data in D10. CH3 port the exports 6~10mA of signals through D12. The user program is written as:

H2U-6AM H2U-6AMR 4-channel analog input/2-channel analog output mixture expansion module

4-channel analog input signal detections and 2-channel analog signal output can be accomplished by the H2U local expansion module. Every input channel will convert 0~20mA of signals into 12bit digital value so that it can be read out by PLC main module. Every output channel has voltage and current signal output ports, and the signal amplitude ranges from -10~10V or 0~10mA. Using the FROM/TO command, the main module accesses the BFM unit in the local expansion module's registers to control the analog output signals.

Parameter specification:

Item		Indicator in AD section
Conversion speed		15ms/channel (regular), 6ms/channel (fast), 1ms/channel (fastest)
Current input signal		0~20mA (input impedance is 250Ω)
Sampling Digital output		Default setting:0 ~ +2047
Dpi		10uA
Accuracy		±1%
Item		Indicator in DA section
Conversion speed		2-channel 2.1mS
Analog output range	Voltage output	-10~10V DC (external load impedance is 2KΩ~1MΩ)
	Current	0~20mA (external load impedance is 500Ω or

	output	smaller)
Digital input		Default setting: -2000 ~ 2000, allowed range: -10000 ~ 10000
Dpi	Voltage output	5mV(10V/2000)
	Current output	10μA(20mA /2000)
Overall accuracy		±1% (overall range of 10V) ±1% (overall range of 20mA)

Module User Interface Table:

Item	Description	
Terminal functions	I+: channel positive current input/output terminal; I-: input/output channel's common terminal; 24+/24-: external 24V input terminal; GND: ground protection	
Indicators light	Local expansion module	PWR: lights up when main module is connected and powred on. COM: flashes rapidly when module is functioning normally; slow flashes when error occurs. 24V: lights up when external 24V is connected.
	Remote expansion module	PWR: 24V normal power supply to the module COM: lights up when module is communicating ERR: lights up when error occurs.
Expansion terminal	Local expansion module	Expansion input terminal uses 26-pin trapezoidal connector and connects to module through flat cables; expansion output terminal uses the same connectors.
	Remote expansion module	Expansion input terminal uses 5-pin plugs. 4-core shielded twisted pair (STP) cable or network cable is recommeded.

Local expansion module's address numbering:

Every time when a PLC main module powers on, it will automatically detect all the connected expansion modules and assign numbers to these special modules and the IO expansion terminal. Users cannot interfer or modify the numbering, unless the connection sequence of the module is changed.

The address numbering system the main module uses on special modules is that, the numbering will start from the module closest to the PLC module and start from #0, #1…#7, and etc. IO expansion module inserted in mid way will not be numbered.

Address numbering for remote special expansion module:

The address numbering system in a remote expansion module is: module communication station number +100, and a maximum number of 63 remote expansion modules can be added. The module station number can be configured by switching the DIP on the Station NO.

If in a module A5 and A4 are ON and others are OFF, the binary address becomes 011000, and K24 for decimal. Therefore, when using the FROM/TO command to program the system, the module's serial number become K24+100, which equals #124.

When DIP switches are modified, besides the matching resistor, Baud rate and addresses will not take effect immedately. The system needs to restart in order to adopt the new configuration parameters. CAN addresses consistancey must be noted and not duplicated addresses are allowed.

Accessing the BFM zone in 6AM(R) Module:

The PLC module accesses the BFM units using the FROM/TO reading/writing command.

An expansion module is equipped with EEPROM storage units, which are used to store BFM configuration settings, such as analog input/output channel's signal types, offset values, gain values, etc. The storing function of these units is performed automatically by the configuration status in the corresponding BFM units.

Every register's bandwidth in a BFM is 16bit (equal to 1Word). The BFM are defined according to the 6AM (R) module and as follows:

BFM	R/W Property	Input channel contents	
#0(E)	WR	Select channel signal mode. Every HEX represents 1 input channel. The highest is ch4, and lowest is ch1: (Default Value=H1111) 1 = 4mA~20mA; corresponding digital output: 0~1000 2 = 0mA~20mA; corresponding digital output: 0~1000 3 = channel closed; 5 = 4mA~20mA; corresponding digital output: 0~10000 6= 0mA~20mA; corresponding digital output: 0~10000	
#1	WR	Channel 1	Average filter constant. Used to determine sample numbers for averaging calculation. Setting range is 1~4096, default value at 8. If rapid sampling is required, value can be set at 1. When BFM#15 changes, it restores to the default value.
#2	WR	Channel 2	
#3	WR	Channel 3	
#4	WR	Channel 4	
#5	R	Channel 1	Input channel sampling value data that has been averaged
#6	R	Channel 2	
#7	R	Channel 3	
#8	R	Channel 4	
#9	R	Channel 1	Present data collected from input channel, which is the instantaneous value before filtering process.
#10	R	Channel 2	
#11	R	Channel 3	
#12	R	Channel 4	
#13~14	-		Reserved
#15	WR	Select ADC rate	0=normal speed, 15ms/channel (default); 1=faster conversion, 6ms/channel; 1000~30000=rapid sampling, corresponds to 1ms~30ms/channel
#16~19	-	Reserved	
#20(E)	WR	1=restore settings. Restore all input channel parameters (BFM#0~BFM#32) to default value (factory setting). Default value=0	
#21(E)	WR	2=offset/gain adjustment prohibited ; 1=offset/gain adjustment permitted (default)	
#22(E)	WR	Operation of the lower 8bits correspond to 4 channels	G4 | O4 | G3 | O3 | G2 | O2 | G1 | O1
		Offset/gain adjustment function. When it is not 0, the module will write the BFM23/24 value in the internal corresponding channel control registers.	
#23(E)	WR	Offset value. Analog input value (0, 1, 2 modes) when digital output is 0. Default value is 400.	
#24(E)	WR	Gain value. Analog input value (0, 1, 2 modes) when digital output is +1000. Default value is 20000.	
#25~26	-	Reserved	

#27	R	6AM module software revision
#28	-	Reserved
#29	R	Input channel error status
#30	R	Expansion module identification code. H2U-6AM (R)'s identification code is K4050.
#31	-	Reserved,Not accessible.

<table>
<tr><th>BFM</th><th>R/W property</th><th colspan="5">Output channel contents</th></tr>
<tr><td>#32(E)</td><td>WR</td><td colspan="5">Select output mode. Every HEX represents 1 output channel. 6AM (R) module selects the highest in the lower 8bits as ch6, and lowest HEX as ch5: (Default Value=H00)

0=-10V~10V; correponding digital output: -2000~2000

1 = 4mA~20mA; correponding digital output: 0~1000

2 = 0mA~20mA; correponding digital output: 0~1000

4=-10V~10V; correponding digital output: -10000~10000

5=4mA~20mA; correponding digital output: 0~10000

6=0mA~20mA; correponding digital output: 0~10000</td></tr>
<tr><td>#33</td><td>WR</td><td>Channel 5</td><td colspan="4" rowspan="2">Channel output value, initial value is 0</td></tr>
<tr><td>#34</td><td>WR</td><td>Channel 6</td></tr>
<tr><td>#35~36</td><td>-</td><td colspan="5">Reserved</td></tr>
<tr><td>#37(E)</td><td>WR</td><td colspan="5">Data preservation mode when PLC is powered down. Every HEX represents 1 output channel. 6AM (R) module selects the highest in the lower 8bits as ch6, and lowest HEX as ch5. When x=0, output before the power down is retained; when x=0, output will be reset to the offset setting.</td></tr>
<tr><td>#38~39</td><td>-</td><td colspan="5">Reserved</td></tr>
<tr><td rowspan="2">#40(E)</td><td rowspan="2">WR</td><td>Operation of the lower 4bits corresponding to 2 channels</td><td>G2</td><td>02</td><td>G1</td><td>01</td></tr>
<tr><td colspan="5">CH6/CH5's offset/gain configuration command. Set up according to HEX bits. Initial value=H0.

0=modification prohibited; 1=modification of EEPROM corresponding data is permitted</td></tr>
<tr><td>#41</td><td>-</td><td colspan="5">Reserved</td></tr>
<tr><td>#42</td><td>WR</td><td>Offset data CH5</td><td colspan="4" rowspan="4">Unit: mV or μA

Initial offset value: 0

Initial gain value: +5000, corresonding mode 0

(0, 1, 2 mode)</td></tr>
<tr><td>#43</td><td>WR</td><td>Gain dta CH5</td></tr>
<tr><td>#44</td><td>WR</td><td>Offset data CH6</td></tr>
<tr><td>#45</td><td>WR</td><td>Gain dta CH6</td></tr>
<tr><td>#46~51</td><td>-</td><td colspan="5">Reserved</td></tr>
<tr><td>#52(E)</td><td>WR</td><td colspan="5">Initial value=0. When 1 is written, all output channel BFM units (BFM#32~BFM#64) will be restored to default values.</td></tr>
<tr><td>#53(E)</td><td>WR</td><td colspan="5">1= output property adjustment (initial value) permitted; 2=output property adjustment prohibited</td></tr>
<tr><td>#54~60</td><td>-</td><td colspan="5">Reserved</td></tr>
<tr><td>#61</td><td>R</td><td colspan="5">Output channel error status</td></tr>
</table>

Status message BFM#29 is explained as follows:

BFM#29 Bit No.	ON State	OFF State
b0:	Error occurs. Any one of b0~b3 is not 0. A/D conversion terminated.	No Error

b1:	Errors in module EEPROM's offset/gain settings	offset/gain data correct
b2:	(impossibility)	Power supply normal
b3:	Module hardware failure	Hardware normal
b10:	Digital output exceeds the range of -2048~2047	Digital output value normal
b11:	Sampling filter constant exceeds 1~4096 range	Sampling filter constant normal
b12:	BFM#21 value setting to K2 is prohibited	BFM#21=K2 permitted
Bit4~7 and bit13~15 in BFM#29 have no definition.		

Status message BFM#61 is explained as follows:

BFM#61 bit no.	ON State	OFF State
b0:	Error occurs. Any one of b0~b3 is not 0. D/A conversion terminated.	No Error
b1:	Errors in module EEPROM's offset/gain settings	offset/gain data correct
b2:	(impossibility)	Power supply normal
b3:	Module hardware failure	Hardware normal
b10:	Digital value input exceeds specified range	Digital output value normal
b12:	BFM#21 value setting to K1 is prohibited	BFM#21=K1
Bit4~7 and bit13~15 in BFM#29 have no definition.		

In the chart, the BFM unit with "(E)" mark is the item stored in EEPROM, which has the power failure preservation feature.

Register modification instruction:

The modifications on BFM#0, #32, and (E) unit will result in the module to perform the write operation on EEPROM. The operation takes certain time and each Word requires approximately 300ms. Therefore, when modifying multiple BFM units is required during the PLC programming process, time delay must be added after every above-mentioned BFM unit. Continuous write operation is not recommended as the write command must be correctly completed.

Input channel BFM area description:

BFM#0 Select input channel

Initialization of input channel. The four default channels are all 4~20mA and are controlled by the hexadecimal HXXXX of BFM#0. The lowest control channel is one and the highest control channel is four. Every character's controlling methods are as follows:

X=1 default range 4~20mA (corresponding number 0~1000)

X=2 default range 0~20mA (corresponding number 0~1000)

X=3 channel closed

X=5 default range 4~20mA (corresponding number 0~10000)

X=6 default range 0~20mA (corresponding number 0~10000)

For example: when BFM#0 is H2235, channel one is 4~20mA; channel two is closed; channel three and four are 4~20mA.

Channels not in use can be either closed or not. Closed channel does not take up conversion time (BFM#15). As channel two in the example is closed, the conversion time for the whole channel become the conversion time of the three channels that are not closed (3×BFM#15).

BFM #1~#4 Input channel average sampling number

After every channel's (BFM#9~#12) corresponding sampling values are accumulated with the sampling number (BFM #1~#4), divide it by the sampling number (BFM #1~#4) and save the result in (BFM #5~#8).

BFM #5~#8 saves input channel average sampling values

BFM #9~#12 saves input channel real-time sampling values

BFM #15 Input channel ADC time

Time required for every channel conversion. Please note that the time required for every data update is the product of BFM#15 multiplying the number of channels that are not closed.

For example, BFM #0 is H3311, BFM #1 is K7, BFM #2 is K6, BFM #15 is K10; so the data update time required for BFM #9 and BFM #10 is BFM#0×BFM#15=2×10=20MS. The time required for BFM #5 is BFM#0×BFM#15×BFM#1=2×10×7=140MS. The time required for BFM #6 is BFM#0×BFM#15×BFM#2=2×10×6=120MS. In the program, the FROM/TO command usually takes more time to function. Therefore, the BFM#5 data collected during the process can be retrieved and read using LDP M8012 FROM K0 K5 D10 K1 command, which has the same result of using LD M8000 FROM K0 K5 D10 K1 command. However, the command initiated with M8000 drive will be read in every scanning cycle, which greatly lengthens the program's scanning cycle.

BFM #20 Input channel restores to factory setting

Modify #20 to 1 can restore to the default value.

BFM #21~#24 Input channel offset/gain's definition and configuration instruction:

H2U-6AM (R)'s input channel has two operation modes. The curve properties are as follows:

Input channel's offset and gain values can be configured individually or collectively. The normal gain value’s setting range is 4~32mA. The normal offset value's setting range is -20~20mA.

Before configuring gain/offset values, BFM#21 must be modified to 1 and then BFM#23/24 can be modified. Every channel's offset/gain BFM#22 is allowed and turned on in order to complete the modification. BFM#21 should be modified to 2 to prevent further changes.

All offset/gain values must be consistent for all channels. One channel cannot have an offset value of 1000 while another has an offset value of 1200.

For example: when BFM#0 is in 1111 mode, the modifying offset and gain values for channel one and two are 5mA and 18mA respectively. Following the procedure below:

Modify BFM#21 to 1; after 300MS transmit K5000 and K18000 to BFM#23 and BFM#24 respectively; after another 300MS, allow gain in BFM#22 and in this case, and the BFM#22 should be 00001111 in binary mode, which the BFM#22 is modified to H000F; modification is completed. At the end, modify BFM#21 to 2 to prevent further changes.

Output channel BFM area description:

BFM#32 Select output channel

Output channel initialization. The two default channel are -10~10V, which is controlled by the hexadecimal HXX of BFM #32. The lowest control channel is five and the second control channel is six. Every character’s controlling method is as follows:

X=0 default range -10~10V (corresponding number -2000~2000)

X=1 default range 4mA~20 mA (corresponding number 0~1000)

X=2 default range0mA~20 mA (corresponding number 0~1000)

X=4 default range -10V~10V (corresponding number -10000~10000)

X=5 default range 4mA ~20 mA (corresponding number 0~10000)

X=6 default range 0mA ~20 mA (corresponding number 0~10000)

For example: when BFM#32 is H10, Channel Five is -10~10V; Channel Six is 4~20mA

BFM #33~#34 Output channel output value

Use the TO command to write in data in BFM #33~#34, which can also control the analog output. Both initial values are 0.

BFM #37 Output channel data retention mode

When the main module is switching from RUN to STOP mode, the last mode of RUN will be retained (X=0), or the offset value (X=1) will be outputted.

For example: when BFM#32 is H00 and BFM#37 is H01, the offset values in both channels are 0.1V. When switching from RUN to STOP, BFM#33 and #34's values are 1500 (7.5V). After STOP has completed, channel 5's output voltage becomes 0.1V, while channel 6's output voltage remain at 7.5V.

BFM #40, BFM #42~#45, BFM #53 Output channel offset/gain definition and setting instruction.

H2U-6AM (R)'s output channel has three working modes. Curve properties are as follows:

Among them, the gain value is the corresponding analog output when digital output value is 1000; the offset value is the corresponding analog output when digital output value is 0. Offset and gain values can be set up individually or collectively. The setting parameter's units are mV (in mode 0) and µA (in mode 1 and 2).

BFM #40 is the offset/gain configuration command. Every HEX digit (composed of 4 binary bits) in hexadecimal controls the prohibition or permission status. Please note that the AD input module is controlled by every binary bit. DA and AD modules use different configuration commands in offset/gain configuration. BFM #10~#13 are the offset/gain configuration values and BFM #53 is the curve property configuration command.

Before offset/gain value are configured, BFM #53 must be set up to 1. Relevant unit values in BFM #42~#45 then are modified accordingly. Finally, write in the operating permission character in BFM #40 unit to complete the modification. BFMn #53 should be set to 2 to prevent further modifications.

For example: when BFM#40 is in mode H00, offset and gain values in channel five must be modified to 0.2V and 5.5V respectively. Follow the procedures below:

Set BFM#53 to 1; after 300MS transmit K200 and K5500 to BFM#42 and BFM#43 respectively; turns on BFM#40 to allow offset/gain value, which is to modify BFM#40 to H0011 to complete the procedure. Set BFM#53 to 2 to prevent further modifications.

BFM #52 Output channle restore to factory setting

Set BFM #52 to 1 to restore to the default value.

Programming Example 1:

Connect one H2U-6AM expansion module to the back of the PLC main module, and module #0 is numbered in sequential order. CH1 terminal needs to collect 4~20mA voltage signal and modify the filtering frequency to 6 to save the collected data in D10. CH5 terminal outputs 6~20mA of current signal through D12. The user program is written as:

H2U -4PT H2U-4PTR 4 Channel temperature detection module

4-way PT100 temperature signal detection can now be realized by incorporating the H2U expansion module , which converts the signal into 12bit digital figures so that it can be read by the PLC main module. Through the FROM/TO command, the main module can access the BFM units in the registers of the expansion module.

Parameter specification:

Item	Indication	Description
Temperature detection sensor	Pt100	
Temperature detecting range	Celsius: -200C to +846C Fahrenheit: 392F to 1554.8F	
Input channel no.	4 channels	
Conversion speed	15ms/channel	

Digital output	12bit: -2000~＋2000	
dpi	0.1℃	
Overal accuracy	±1% of all range	

Special expansion module's address numbering:

The address numbering system the main module uses on special modules is that, the numbering will start from the module closest to the PLC module and start from #0, #1…#7, and etc. IO expansion module inserted in mid way will not be numbered.

Address numbering for remote special expansion module:

The address numbering system in a remote expansion module is: module communication station number +100, and a maximum number of 63 remote expansion modules can be added. The module station number can be configured by switching the DIP on the Station NO.

If in a module A5 and A4 are ON and others are OFF, the binary address becomes 011000, and K24 for decimal. Therefore, when using the FROM/TO command to program the system, the module's serial number become K24+100, which equals #124.

When DIP switches are modified, besides the matching resistor, Baud rate and addresses will not take effect immedately. The system needs to restart in order to adopt the new configuration parameters. CAN addresses uniqueness must be noted and not duplicated addresses are allowed.

Accessing the BFM zone in 4PT (R) module:

The PLC main module reads the digital AD conversion results through 4PT(R) module's register cache units (BFM zone). Module condition is configured through modifying in specific BFM zones. The PLC module accesses the BFM units using the FROM/TO reading/writing command.

Every register's bandwidth in the BFM zone is 16bit (equals to 1Word). According to the 4PT(R) module, the BFM zone is defined as the following chart:

BFM	R/W property	Contents	
#0	-	Reserved	
#1	WR	Channel 1	Average filter constant. Used to determine sample numbers for averaging calculation. Setting range is 1~4096, default value at 8. If rapid sampling is required, value can be set at 1.
#2	WR	Channel 2	
#3	WR	Channel 3	
#4	WR	Channel 4	
#5	WR	Channel 1	Average temperature from CH1 to CH4 under 0.1C unit
#6	WR	Channel 2	
#7	WR	Channel 3	
#8	WR	Channel 4	
#9	WR	Channel 1	Current temperature from CH1 to CH4 under 0.1C unit
#10	WR	Channel 2	
#11	WR	Channel 3	
#12	WR	Channel 4	
#13	WR	Channel 1	Average temperature from CH1 to CH4 under 0.1F unit
#14	WR	Channel 2	
#15	WR	Channel 3	
#16	WR	Channel 4	
#17	WR	Channel 1	Current temperature from CH1 to CH4
#18	WR	Channel 2	

#19	WR	Channel 3	under 0.1F unit
#20	WR	Channel 4	
#21~26	-	Reserved	
#27	R	4PT module software edition	
#28	R/W	Digital range error latch (thermal resistance disconnection can be detected)	
#29	R	Error	
#30	R	Expansion module identification code. H2U-4PT (R)'s identification code is K2040.	
#31	-	Reserved, not accessible	

Status message BFM#28 is described as:

b15 to b8	b7	b6	b5	b4	b3	b2	b1	b0
Unused	High	Low	High	Low	High	Low	High	Low
	CH4		CH3		CH2		CH1	

Low: when temperature measured is lower than the minimum detectable temperature, latch ON.

High: when temperature measured is higher than the maximum detectable temperature, latch ON.

Temperature returns to normal range after error occurs. Error will still be latched in BFM#28.

Use TO command to write K0 into BFM28 or turn the power off to clear errors.

Status message BFM#29 is described as:

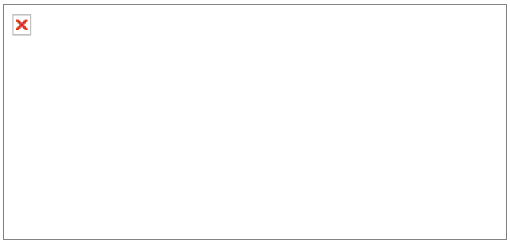
BFM#29 No.	ON state	OFF state
b0:	Error occurs. When any of b0~b3 is not 0, A/D conversion stops.	No erros
b1:	Reserved	Reserved
b2:	(impossibility)	Power supply normal
b3:	Module hardware failure	Hardware normal
b10:	Digital output exceed designated range	Digital output normal
b11:	Sampling filter constant exceeds the range of 1~4096	Sample filter constant normal
b12:	Reserved	Reserved
Bit4~7, bit13~15 in BFM#29 has no definition.		

NOTE: when the external 24V power supply of the local expansion module has a power failure, PLC main module's system mark M8067 will be reset,error code D8067=K6708. The mark should be regularly checked for immediate notice.

Programming Example:

Connect one H2U-4PT expansion module to the back of the PLC main module, and module #0 is numbered in sequential order. CH1-CH4 terminals need to collect PT100 voltage signal and modify the filtering frequency to 6. Save the collected data from four channels in D10, D11, D12, and D13 respectively. The user program is written as:

In the example, if H2U-4PTR remote module is used instead, the CAN station number becomes 1. The programming for this example will be as follows:



H2U-4TC H2U-4TCR 4 Channel temperature detection module

4-way K/J type temperature signal detection can be realized by incorporating the H2U expansion module , which converts the signal into 12bit digital figures so that it can be read by the PLC main module. Through the FROM/TO command, the main module can access the BFM units in the registers of the expansion module.

Parameter specification:

Item	°C	°F
Temperature detection sensor	K/J type thermocouple	
Temperature detection range	K type: -100C to +1200C J type: -100C to +600C	K type: -148F to +2192F J type: -148F to +1112F
Input channel no.	4 channels	
Conversion speed	240ms/4 channel	
Digital output	K type: -1000 to +12000 J type: -1000 to +6000	K type: -1480 to +21920 J type: -1480 to +11120
dpi	0.1℃	0.1F
Overall accuracy	±0.5% of overall range +1 ℃	
I/O points used	8 points	
	Optocoupler is used to isolate between analog and digital	

Isolation	circuits; DC/DC is used to isolate between analog circuit and external power supplies; No isolation is required between analog input signal channels.

Special expansion module's address numbering:

The address numbering system the main module uses on special modules is that, the numbering will start from the module closest to the PLC module and start from #0, #1…#7, and etc. IO expansion module inserted in mid way will not be numbered.

The address ID of the remote special extension module:

In the remote extension module, remote module address is: No. +100 communication station, allowing up to 63 remote extension modules. Through toggle the DIP switch on the Station NO, to set the station number of the modules.

If A5, A4 on one module are both ON, others are OFF, the binary address is: 011 000, decimal K24, then when we use the FORM / TO instruction program the module is K24 +100, which is # 124.

If you change the DIP switch, in addition to matching resistor, the baud rate and address can not immediately take effect, the system may be used to re-power on before setting the new parameters.

Accessing the BFM zone in 4TC (R) module**:**

The PLC main module reads the digital AD conversion results through 4TC(R) module's register cache units (BFM zone). Module condition is configured through modifying in specific BFM zones. PLC main module access the BFM module by read and write instructions FROM / TO.

Every register's bandwidth in the BFM zone is 16bit (equals to 1Word). According to the 4TC(R) module, the BFM zone is defined as:

BFM	R/W property	Contents	
#0	WR	Select output mode. Every HEX bit represents 1 input channel. The highest bit is Ch4 and the lowest bit is Ch1: (default is H0000) 0=K Type;1=J Type;	
#1	WR	Channel 1	Average filter constant, that is, the number of samples for the average calculation, setting range 1 to 256, the default value of 8. To high speed collection, can be set to 1.
#2	WR	Channel 2	
#3	WR	Channel 3	
#4	WR	Channel 4	
#5	WR	Channel 1	Average temperature from CH1 to CH4 under 0.1C unit
#6	WR	Channel 2	
#7	WR	Channel 3	
#8	WR	Channel 4	
#9	WR	Channel 1	Current temperature from CH1 to CH4 under 0.1C unit
#10	WR	Channel 2	
#11	WR	Channel 3	
#12	WR	Channel 4	
#13	WR	Channel 1	Average temperature from CH1 to CH4 under 0.1F unit
#14	WR	Channel 2	
#15	WR	Channel 3	
#16	WR	Channel 4	
#17	WR	Channel 1	

#18	WR	Channel 2	Current temperature from CH1 to CH4 under 0.1F unit
#19	WR	Channel 3	
#20	WR	Channel 4	
#21~26	-	Reserved	
#27	R	4TC module software version	
#28	R/W	Digital range error latch (thermal resistance disconnection can be detected)	
#29	R	Error state	
#30	R	Expansion module identification code. H2U-4TC(R)'s identification code is K2030.	
#31	-	Reserved, not accessible	

Status message BFM#28 is described as:

b15 to b8	b7	b6	b5	b4	b3	b2	b1	b0
Unused	High	Low	High	Low	High	Low	High	Low
	CH4		CH3		CH2		CH1	

Low: when temperature measured is lower than the minimum detectable temperature, latch ON.

High: when temperature measured is higher than the maximum detectable temperature, latch ON.

Temperature returns to normal range after error occurs. Error will still be latched in BFM#28.

Use TO command to write K0 into BFM28 or turn the power off to clear errors.

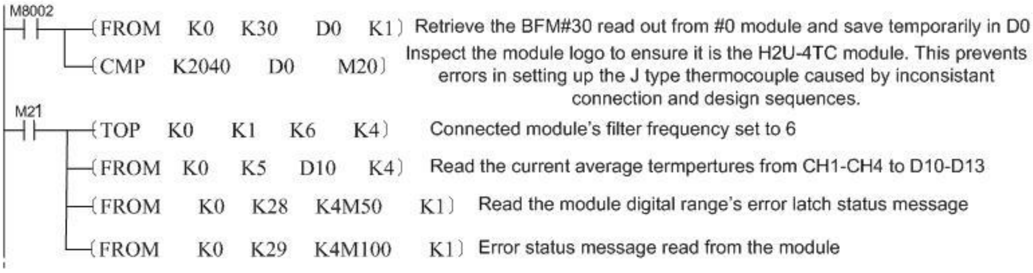
NOTE: when the external 24V power supply of the local expansion module has a power failure, PLC main module's system mark M8067 will be reset,error code D8067=K6708. The mark should be regularly checked for immediate notice.

Status message BFM#29 is described as:

BFM#29 Bit No.	ON state	OFF state
b0:	Error occurs. When any of b0~b3 is not 0, A/D conversion stops.	No error
b1:	Reserved	Reserved
b2:	(impossibility)	power supply normal
b3:	module hardware failure	Hardware normal
b10:	Digital output exceed designated range	Digital output normal
b11:	Sampling filter constant exceeds the range of 1~256	Sample filter constant normal
b12:	Reserved	Reserved
Bit4~7, bit13~15 in BFM#29 has no definition.		

Programming Example:

Connect one H2U-4TC expansion module to the back of the PLC main module, and module #0 is numbered in sequential order. CH1-CH4 terminals need to collect the temperature value of the J type thermocouple. Asked to change filter number as 6. The data collected from two channels are stored in D10, D11. User program written as follows:



In the example, if H2U-4TCR remote module is used instead, the CAN station number becomes 1. The programming for this example will be as follows:

Quickly search a series of MD320 transducer functional code

Following is instruction of character in Functional code:

"☆": The parameter can be modified whenever state of transducer is stop or run;

"★": The parameter can not be modified when state of transducer is run;

"●": The parameter is actual detect record key,can not be modified ;

"*": The parameter is "manufacturer setting",only can be set by manufacturer,forbidden user to operate;

Setting relation graph of functional code in common use

Group F0 Group of basic function

Functional code	Name	LED menu display	The range of setting	Min. unit	Initialization	Modify illuminate
Group F0 group of basic function						
F0-00	model display	model display	1: G model (constant torque loade model） 2: P model (type of wind machine、water pump load model)	1	relate to model	●

F0-01	control mode	control mode	0: zero velocity sensor vector control(SVC) 1: velocity sensor vector control(VC) 2: V/F control	1	0	★	
F0-02	choosecommand headstream	choosecommand headstream	0: command channels of manipulation faceplate run (LED OFF); 1: command channels of terminal(LED ON); 2: command channels of com (LED Flash)	1	0	☆	
F0-03	choose host frequency headstream X	choose frequency headstream X	0: adjust number setting UP,DOWN (cann't memory) 1: adjust number setting UP,DOWN (memory) 2: AI1 3: AI2 4: AI3 5: PULSE setting (DI5) 6: velocity of many segment 7: PLC 8: PID 9: communication	1	1	★	

			give			
F0-04	choose assistant frequency headstream Y	choose frequency headstream Y	0: number setting UP,DOWN(cann't memory) 1: number setting UP,DOWN (memory) 2: AI1 3: AI2 4: AI3 5: PULSE pulse setting (X5) 6: velocity of many segment 7: PLC 8: PID 9: communication give	1	0	★
F0-05	choose range of assistant frequency headstream Y	choose range of Y	0:relative to max. frequency 1:relative to frequency headstream X	1	0	☆
F0-06	range of assistant frequency headstream Y	range of frequency headstream Y	0%~100%	1%	100%	☆
			0: host frequency headstream X 1: host frequency			

F0-07	choose frequency headstream	choose frequency headstream	headstream X+assistant frequency headstream Y 2: switch host frequency headstream X to assistant frequency headstream Y 3: switch host frequency headstream Xto (host frequency headstream X+assistant frequency headstream Y) 4: switch assistant frequency headstream Y to (host frequency headstream X+assistant host frequency headstream Y)	1	0	☆
F0-08	preset frequency	preset frequency	0.00Hz~max. frequencyF0-10	0.01Hz	50.00Hz	☆
F0-09	run direction	run direction	0: same direction 1: opposite direction	1	0	★
F0-10	max. frequency	max. frequency	50.00Hz~300.00Hz	1	50Hz	★
			0: F0-12 setting			

F0-11	upper limit frequency headstream	upper limit frequency headstream	1: AI1 2: AI2 3: AI3 4: PULSE setting 5: communication give	1	0	★
F0-12	upper limit frequency	upper limit frequency	lower limit frequency F0-14~max. frequencyF0-10	0.01Hz	50.00Hz	☆
F0-13	upper limit frequency bias	upper limit frequencybias	0.00Hz~max. frequency F0-10	0.01Hz	0.00Hz	☆
F0-14	lower limitfrequency	lower limit frequency	0.00Hz~upper limit frequency F0-12	0.01Hz	0.00Hz	☆
F0-15	carrier frequency	carrier frequency	0.5kHz~16.0kHz	0.1kHz	relate to model	☆
F0-16	adjust choose from carrier frequency	adjust choose from carrier frequency	0: fixation PWM,adjust carrier frequency temperature invalid 1: random PWM,adjust carrier frequency temperature invalid 2: fixation PWM,adjust carrier frequency temperature valid 3: random PWM,adjust carrier	1	2	☆

			frequency temperature valid			
F0-17	accelerative time1	accelerative time1	0.0s~6500.0s	0.1s	20.0s	☆
F0-18	decelerated time1	decelerated time1	0.0s~6500.0s	0.1s	20.0s	☆

Group F1 Electric machinery parameter

Group F1 **electric machinery parameter**						
F1-00	choose type of electric machinery	choose type of electric machinery	0: general asynchronism electric machinery 1: frequency conversion asynchronism electric machinery 2: permanent-magnet synchronization electric machinery	1	0	★
F1-01	rated power	rated power	0.4kW~1000.0kW	0.1kW	affirm model	★
F1-02	rated voltage	rated voltage	0V~440V	1V	380V	★
F1-03	rated electric current	rated electric current	0.01A~ 655.35A	0.01A	affirm model	★
F1-04	rated frequency	rated frequency	0.00max.~ frequency	0.01Hz	50.00Hz	★
F1-05	rated revolutions per minute	rated revolutions per minute	0rpm~30000rpm	1rpm	1460rpm	★
F1-06	stator resistance	stator resistance	0.001Ω~65.535Ω	0.001Ω	affirm model	☆
F1-07	rotor resistance	rotor resistance	0.001Ω~65.535Ω	0.001Ω	affirm model	☆
F1-08	leakage inductance	leakage inductance	0.01mH~655.35mH	0.01mH	affirm model	☆
F1-09	mutual inductance	mutual inductance	0.1mH~6553.5mH	0.1mH	affirm model	☆
F1-10	no carry electric current	no carry electric current	0.01A~650.00A	0.01A	affirm model	☆
F1-11	choose tuning	choose tuning	0: no operation 1: stillness tuning 2: tuning integrity	1	0	★

Group F2 Vector control parameter

F2 Vector control parameter						
F2-00	velocity loop proportion gain 1	velocity loopP1	0~100	1	30	☆
F2-01	velocity loop integral time1	velocity loopI1	0.01s~10.00s	0.01s	0.50s	☆
F2-02	switch frequency	switch frequency	0.00~F2-05	0.01Hz	5.00Hz	☆

	1	1				
F2-03	velocity loop proportional gain 2	velocity loop P2	0~100	1	20	☆
F2-04	velocity loop integral time 2	velocity loop I2	0.01s~10.00s	0.01s	1.00s	☆
F2-05	switch frequency 2	switch frequency 2	F2-02~max. frequency	0.01Hz	10.00Hz	☆
F2-06	rotate dispersion compensator coefficient	rotate dispersion coefficient	50%~200%	1%	100%	☆
F2-07	constant of velocity loop filter time	velocity loop filter	0.000s~0.100s	0.001s	0.000s	☆
F2-08	torque control	torque control	0: invalid 1: valid	1	0	☆
F2-09	torque upper limit headstream	torque upper limit headstream	0: F2-10 1: AI1 2: AI2 3: AI3 4: PULSE setting 5: communication give range of simulation input is F2-10	1	0	☆
F2-10	torque upper limit	torque upper limit	0.0%~200.0%	0.1%	150.0%	☆
F2-11	encoder buffer	encoder buffer	1~65535	1	1024	★
F2-12	obligate	obligate	0~65535	1	0	★

Group F3 V/F control parameters

F3 V/F control parameters						
F3-00	V/F curve setting	V/F curve setting	0: beeline V/F curve 1: many points V/F curve 2: square V/F curve	1	0	★
F3-01	advance torque	advance torque	0.0: (Auto)0.1% ~30.0%	0.1%	1.0%	☆
F3-02	advance torque end frequency	advance torque frequency	0.00Hz~Max. frequency	0.01Hz	50.00Hz	★
F3-03	V/F frequency point 1	V/F frequency1	0.00Hz~electric machinery rated frequency	0.01Hz	0.00Hz	★
F3-04	V/F voltagepoint 1	V/F voltage 1	0.0%~100.0%	0.1%	0.0%	★
F3-05	V/F frequencypoint 2	V/F frequency 2	0.00Hz~electric machinery rated frequency	0.01Hz	0.00Hz	★
F3-06	V/F voltagepoint 2	V/F voltage 2	0.0%~100.0%	0.1%	0.0%	★
F3-07	V/Ffrequency point 3	V/F frequency	0.00Hz~electric machinery rated	0.01Hz	0.00Hz	★

		3	frequency			
F3-08	V/F voltagepoint 3	V/F voltage 3	0.0%~100.0%	0.1%	0.0%	★
F3-09	slip compensative coefficient	slip compensative coefficient	0.0%~200.0%	0.1%	0.0%	☆
F3-10	choose AVR	choose AVR	0: invalid 1: valid whole course 2: only when decelerating invalid	1	2	☆
F3-11	surge restrain gain	surge restrain gain	0~100	1	affirm model	☆

Group F4 Import terminal

F4 import **terminal**						
F4-00	DI1 terminal function choose	DI1 terminal choose	0: No function 1: Forward run (FWD) 2: Reverse run(REV) 3: Three thread model running control 4: Forward jog (FJOG) 5: Reverse jog (RJOG) 6: Terminal UP 7: Terminal DOWN 8: Free parking 9: Malfunction reset (RESET) 10: Run pause 11: External malfunction ON input 12: Ssegments of velocity terminal 1 13: Segments of velocity terminal 2 14: Segments of velocity terminal 3 15: Segments of velocity terminal 4 16: Choose acceleration or deceleration terminal 1 17: Choose acceleration or deceleration terminal	1	1	★
F4-01	DI2 terminal functionchoose	DI2 terminal choose		1	4	★
F4-02	DI3 terminal function choose	DI3 terminal choose		1	9	★
F4-03	DI4 terminal function choose	DI4 terminal choose		1	12	★
F4-04	DI5 terminal function choose	DI5 terminal choose		1	13	★
F4-05	DI6terminal function choose(on IO extended card)	DI6 terminal choose		1	0	★
F4-06	DI7 terminal function choose(on IO extended card)	DI7 terminal choose		1	0	★
F4-07	DI8terminal function choose(on IO	DI8 terminal		1	0	★

	extended card)	choose	2			
F4-08	DI9terminal function choose(on IO extended card)	DI9 terminal choose	18: Switch of frequency stream 19: UP/DOWN reset (terminal, keyboard)	1	0	★
F4-09	DI10 terminal function choose(on IO extended card)	DI10 terminal choose	20: Command run swich terminal 21: Acceleration or deceleration forbiddance 22: PID pause 23: PLC state restore 24: Swing frequency pause 25: Counter import 26: Counter restore 27: Length count import 28: Length restore 29: Forbid torque control 30: PULSE(pulse) frequency import (only take effect DI5) 31: Reserve 32: Direct current brake order 33: External malfunction OFF import 34: Frequency setting take effect terminal (Don't set the terminal function ,format is available) If set function of the terminal,when frequency modified,controled modify availd time according to the terminal. 35: Direction of PID take effect in reverse terminal If the terminal availability,PID effect direction in reverse to setting direction of FA-03. 36: External parking terminal When keyboard control，can use the terminal parking, quite	1	0	★

			to key STOPon keyboard. 37: Control order swich terminal 2. Use to swich from terminal control to communicate control，the terminal available .If F0-02 is terminal control,switch to communicate control.If F0-02is communicate control,switch toterminal control.			
			38: PID integral pause terminal If the terminal availability,PID integral function paused,but proportional and derivative adjusted still usefully. 39: Frequency stream X and advance setting frequency switch terminal If the terminal availability,frequency stream X replaced by advance setting frequency(F0-08). 40:Frequency stream Y and advance setting frequency switch terminal If the terminal availability,frequency stream Y replaced by advance setting frequency(F0-08).			
F4-10	DI terminal filter time	DI filter time	1~10	1	4	☆
F4-11	terminal command mode	terminal command mode	0: Two thread model 1 1: Two thread model 2 2: Tthree thread model 1 3: Three thread model 2	1	0	★
F4-12	terminalUP/DOWN change rate	terminalUP/DOWNchange rate	0.01 Hz/s~100.00Hz/s	0.01Hz/s	1.00Hz/s	☆
F4-13	AI1Min. import	AI1Min. import	0.00V~10.00V	0.01V	0.00V (waiting be affirmed)	☆
F4-14	AI1 Min. import correspondence setting	AI1 Min. setting	-100.0%~100.0%	0.1%	0.0%	☆

F4-15	AI1Max. import	AI1 Max. import	0.00V~10.00V	0.01V	10.00V	☆
F4-16	AI1 Max. import correspondence setting	AI1 Max. setting	-100.0%~100.0%	0.1%	100.0%	☆
F4-17	AI1 import filter time	AI1 filter time	0.00s~10.00s	0.01s	0.10s	☆
F4-18	AI2 Min. import	AI2 Min. import	0.00V~10.00V	0.01V	0.00V	
F4-19	AI2 Min. import correspondence setting	AI2 Min. setting	-100.0%~100.0%	0.1%	0.0%	☆
F4-20	AI2 Max. import	AI2 Max. import	0.00V~10.00V	0.01V	10.00V	☆
F4-21	AI2 Max. import correspondence setting	AI2 Max. setting	-100.0%~100.0%	0.1%	100.0V	☆
F4-22	AI2 import filter time	AI2 filter time	0.00s~10.00s	0.01s	0.10s	☆
F4-23	AI3 Min. import (on IO extended card)	AI3 Min. import	0.00V~10.00V	0.01V	0.00V	☆
F4-24	AI3 Min. import correspondence setting(on IO extended card)	AI3 Min setting	-100.0%~100.0%	0.1%	0.0%	☆
F4-25	AI3 Max. import (on IO extended card)	AI3 Max. import	0.00V~10.00V	0.01V	10.00V	☆
F4-26	AI3 Max. import correspondence setting(on IO extended card)	AI3 Max. setting	-100.0%~100.0%	0.1%	100.0%	☆
F4-27	AI3 import filter time (on IO extended card)	AI3 filter time	0.00s~10.00s	0.01s	0.10s	☆
F4-28	PULSE (pulse) import Min. frequency	Min. frequency of pulse	0.00kHz~50.00kHz	0.01kHz	0.00kHz	☆
F4-29	PULSE (pulse) import Min. frequency correspondence setting	setting Max. pulse	-100.0%~100.0%	0.1%	0.0%	☆
F4-30	PULSE (pulse) import Max.	Max. frequency of pulse	0.00kHz~50.00kHz	0.01kHz	50.00kHz	☆

	frequency					
F4-31	PULSE (pulse) import Max. frequency correspondence setting	setting Max. pulse	-100.0%~100.0%	0.1%	100.0%	☆
F4-32	PULSE (pulse) import filtertime	pulse filter time	0.00s~10.00s	0.01s	0.10s	☆

Group F5 Export terminal

Group F5 export terminal						
F5-00	FM choose terminal export model	FM terminal model	0: Pulse export(FMP) 1: Open-collector swiching value export(FMR)	1	0	☆
F5-01	FMR export choose	FMR export choose	0: No export 1: Transducer running 2: Malfunction export 3: Frequency horizontal detect FDT export 4: Frequency arrived 5: Zero speed running 6: Eelectric machinery over loading alarm in advance 7: Transducer over loading alarm in advance 8: B enacted numerical value arrived 9: Be appointed numerical value arrived 10: Length arrivedt 11: PLC circle completed 12: Runtime arrived 13: Frequency limiting 14: Torque limiting 15: Run in train 16: AI1>AI2 16: Reserve 17: Upper limit frequency arrived	1	0	★
F5-02	control board relay**(T/A-T/B-T/C)** export choose	control board relay **RELAY1**export choose		1	2	☆
F5-03	extended card relay **(P/A-P/B-P/C)**export choose	extended card relay **RELAY2**export choose		1	0	☆
F5-04	DO1 export choose	DO1 export choose		1	1	☆

F5-05	extended card DO2 export choose	DO2 export choose	18: Lower limit frequency arrived 19: Over low voltage state export 20: Communicate setting	1	4	☆
F5-06	FMP export choose	FMP export choose	0: Run frequency 1: Setting frequency 2: Export electric current 3: Export torque 4: Export power 5: Export voltage 6: PULSE import 7: AI1 8: AI2 9: AI3(extended card) 10: Length 11: Take count of 12: Communicate setting 13: Reserve 14: Reserve 15: Reserve 16: Waiting be affirmed	1	0	☆
F5-07	AO1 export choose	AO1 export choose			0	☆
F5-08	extended card AO2export choose	AO2 export choose			1	☆
F5-09	FMP export Max. frequency	FMP Max. frequency	0.1kHz¡«50.0kHz	0kHz	50.0kHz	☆
F5-10	AO1 Null shift coefficient	AO1Null shift	-100.0%¡«100.0%	0.1%	0.0%	☆
F5-11	AO1 gain	AO1 gain	-10.00¡«10.00	0.01	1.00	☆
F5-12	AO2 Null shift coefficient (on IO extended card)	AO2 Null shift	-100.0%¡«100.0%	0.1%	0.0%	☆
F5-13	AO2 gain (on IO extended card)	AO2 gain	-10.00¡«10.00	0.01	1.00	☆

Group F6 RUN/STOP control

Group F6 RUN/STOP control						

F6-00	startup mode	startup mode	0: directly startup 1: velocity track and startup	1	0	☆
F6-01	rotate speed track mode	rotate speed track mode	0: start from frequency stop 1: start from zero speed 2: start from Max. frequency	1	0	★
F6-02	rotate speed track speed	rotate speed track speed	1~100	1	20	☆
F6-03	startup frequency	startup frequency	0.00 Hz~10.00Hz	0.01Hz	0.00Hz	☆
F6-04	startup frequency hold time	startup holdtime	0.0s~36.0s	0.1s	0.0s	★
F6-05	startup direct current trig electric current	startup trig electric current	0%~100%	1%	0%	★
F6-06	startup direct current trig time	startup trig time	0.0s~36.0s	0.1s	0.0s	★

F6-07	accelerative and decelerative mode	accelerative and decelerative mode	0: beeline accelerative and decelerative 1: S curve accelerative and decelerative	1	0	★
F6-08	S curve start segment time	S curve start segment	0.0%~40.0%	0.1%	30.0%	★
F6-09	S curve end segment time	S curve end segment	0.0%~40.0%	0.1%	30.0%	★
F6-10	Stop mode	Stop mode	0: decelerative parking 1: free parking	1	0	☆
F6-11	Stop direct current start trig frequency	Stop trig frequency	0.00Hz~Max. frequency	0.01Hz	0.00Hz	☆
F6-12	Stop direct current trig wait time	Stop trig wait	0.0s~36.0s	0.1s	0.0s	☆
F6-13	Stop direct current trigelectric	Stop trigelectric current	0%~100%	1%	0%	☆

	current					
F6-14	Stop direct current trigtime	Stop trig time	0.0s~36.0s	0.1s	0.0s	☆
F6-15	trig Using rate	trig Using rate	0%~100%	1%	100%	☆

Group F7 Keyboard and display

GroupF7 keyboard and display						
F7-00	**LCD language select**	**language select**	**0: Chinese** **1: English**	**1**	**0**	☆
F7-01	**MF.K function select**	**MF.K function select**	**0: MF.K invalidation** **1: manipulate panel command channeland long-distance command channel(terminal command channel or com communication command channel) switch** **2: forward or reverse switch** **3: forward jog**	**1**	**0**	★
F7-02	**STOP/RESETfunction**	**STOP function**	**0: Only avail when keyboard control** **1: When terminal control,STOP means function of stop machine availability** **2: When terminal control,STOP means function of malfunction reset availability** **3: When terminal control,STOP means function of stop machine and malfunction reset function are availability.**	**1**	**0**	☆
F7-03	**QUICK parameter lock**	**parameter lock**	**0: QUICK parameter lock invalidation** **1: QUICK parameter lock validation**	**1**	**0**	☆
F7-	**LED running**	**Running**				

04	display parameter	display	0~65535	1	0	☆
F7-05	LED stop displayparameter	Stop display	1~65535	1	255	☆
F7-06	load speed display coefficient	load speed coefficient	0.0001~6.5000	0.0001	1.0000	☆
F7-07	radiator temperature 1	radiator temperature1	0.0℃~ 100℃	1℃	-	●
F7-08	radiator temperature 2	radiator temperature 2	0.0℃~ 100℃	1℃	-	●
F7-09	cumulate running time	cumulate running time	0h~65535h	1	-	●
F7-10	Software version number 1	Software version number 1	-	-	-	●
F7-11	Software version number 2	Software version number 2	-	-	-	●

Group F8 Assistant function

GroupF8 assistant function						
F8-00	jog run frequency	jog runfrequency	0.00Hz~Max. frequency	0.01Hz	2.00Hz	☆
F8-01	jog accelerative time	jog accelerative time	0.0s~6500.0s	0.1s	20.0s	☆
F8-02	jog deceleration time	jog deceleration time	0.0s~6500.0s	0.1s	20.0s	☆
F8-03	accelerative time 2	accelerative time 2	0.0s~6500.0s	0.1s	20.0s	☆
F8-04	deceleration time 2	deceleration time 2	0.0s~6500.0s	0.1s	20.0s	☆
F8-05	accelerative time 3	accelerative time 3	0.0s~6500.0s	0.1s	20.0s	☆
F8-06	deceleration time 3	deceleration time 3	0.0s~6500.0s	0.1s	20.0s	☆
F8-07	accelerative time 4	accelerative time 4	0.0s~6500.0s	0.1s	20.0s	☆
F8-08	deceleration time 4	deceleration time 4	0.0s~6500.0s	0.1s	20.0s	☆
F8-09	leap frequency 1	leap frequency 1	0.00 Hz~Max. frequency	0.01Hz	0.00Hz	☆
F8-10	leap frequency 2	leap frequency 2	0.00 Hz¡«Max. frequency	0.01Hz	0.00Hz	☆
F8-11	leap frequency range	leap frequency range	0.00 Hz¡«Max. frequency	0.01Hz	0.01Hz	☆
F8-12	forwarda and reversal stagnant area time	forwarda and reversal stagnant area time	0.0s¡«3000.0s	0.1s	0.0s	☆
F8-13	reversal control	reversal control	0: allowed reversal 1: forbidden reversal	1	0	☆
F8-14	action when setting frequency lower than	give effect to lower limit frequency	0: Run with lower limit frequency 1: Stop 2: Run with zero	1	0	☆

	lower limit frequency		speed			
F8-15	droop control	droop control	0.00Hz¡«10.00Hz	0.01Hz	0.00Hz	☆
F8-16	over modulate enable	over modulate enable	0: over modulate invalidation 1: over modulate availability	1	1	☆
F8-17	set runtime	set runtime	0h~65535h	1h	65535h	☆
F8-18	startup protect selected	startup protect selected	0: Nno protect 1: Protect	1	0	☆
F8-19	frequencydetect value (FDTlevel)	FDT level	0.00~Max. frequency	0.01Hz	50.00Hz	☆
F8-20	frequency detect lag value (FDTlag)	FDT lag	0.0%~100.0% (FDT level)	0.1%	5.0%	☆
F8-21	frequency arrived detect width	frequencyarrived width	0.0%~100.0% (Max. frequency)	0.1%	0.0%	☆
F8-22	electrify short circuit to earth protect detect	electrify short circuit to earth detect	0: Invalidation 1: Availability	1	1	☆
F8-23	runtime to act choose	runtime to act choose	0: Continue running 1: Stop	1	0	★

Group F9 Malfunction and safeguard

Group F9 **malfunction and safeguard**						
F9-00	selectedelectric machinery over loading protect	selected over loading protect	0: forbiddence 1: allowed	1	1	☆
F9-01	electric machinery over loading protect gain	over loading protectcoefficient	0.20~10.00	0.01	1.00	☆
F9-02	electric machinery over loading alarm in advancecoefficient	over loading alarm in advancecoefficient	50%~100%	1%	80%	☆
F9-03	over-voltage lose speed gain	over-voltage lose speedgain	0(no over-voltage lose speed)~100	1	0	☆
F9-04	over-voltage lose speed protect voltage	over-voltage lose speed point	120%~150%	1%	130%	☆
F9-05	over-current lose speed gain	over-current lose speed gain	0~100	1	20	☆
F9	over-current lose speed	over-current				

06	protect electric current	lose speed point	100%~200%	1%	150%	☆
F9-07	instant stop null stop function	instant stop null stop function	0: forbiddence 1: allowed	1	0	☆
F9-08	instant stop null stop frequency rate descend rate	instant stop null stop frequency ratedescend rate	0.00Hz/s~Max. frequency /s	0.01Hz/s	10.00Hz/s	☆
F9-09	malfunction auto reset times	malfunction reset times	0~3	1	0	☆
F9-10	when malfunctionauto reset,malfunction selected relay action (T/A-T/B-T/C)		0: no action 1: action	1	0	☆
F9-11	malfunction auto reset spacing time	malfunctionreset spacing	0.1s~100.0s	0.1s	1.0s	☆
F9-12	import lack phase protect choose	import lack phase choose	0: forbiddence 1: allowed	1	1	☆
F9-13	export lack phase protect choose	export lack phase choose	0: forbiddence 1: allowed	1	1	☆
F9-14	frist malfunctiontype	second malfunction type 1	0: No malfunction 1: contravariant unit protect (ERR01) 2: accelerate over- current (ERR02) 3: decelerate over-current (ERR03) 4: constant speed over-current (ERR04) 5: accelerate over-voltage (ERR05) 6: decelerate over-voltage (ERR06) 7: constant speed over-voltage (ERR07) 8: control power malfunction	-	-	●

			(ERR08) 9: over low voltage malfunction (ERR09) 10: transducer over loading (ERR10) 11: electric machinery over loading (ERR11)			
F9-15	second malfunction type	second malfunction type 2	12: import lack phase(ERR12) 13: export lack phase(ERR13) 14: radiator over-heat (ERR14) 15: external malfunction (ERR15) 16: communication malfunction (ERR16)	-	-	•
F9-16	the last malfunction type	third malfunction type	17: contactor malfunction (ERR17) 18: electric current detect malfunction (ERR18) 19: electric machinery tune malfunction (ERR19) 20: encoding disk malfunction (ERR20) 21: data overflow (ERR21) 22: transducer hardware malfunction (ERR22) 23: the malfunction which is electric machinery short circuit to earth(ERR23) 24: reserve (ERR24)	-	-	•
F9-17	frequency when malfunction	frequency when malfunction	-	-	-	•

F9-18	electric current when malfunction	electric current when malfunction	-	-	-	•
F9-19	generatrix voltage when malfunction	generatrix voltage when malfunction	-	-	-	•
F9-20	export terminal when malfunction	exportterminal when malfunction	-	-	-	•
F9-21	export terminal when malfunction	exportterminal when malfunction	-	-	-	•

Group FA PID function

Group FA **PID function**						
FA-00	PID setting headstream	PID setting headstream	0: FA-01 1: AI1 2: AI2 3: AI3 4: PULSE setting (DI5) 5: Communication setting	1	0	☆
FA-01	PID keyboard setting	PID setting	0.0%~100.0%	0.1	50.0%	☆
FA-02	PID feedback headstream	PID feedback headstream	0: AI1 1: AI2 2: AI3 3: AI1-AI2 4: PULSE setting (DI5) 5:Communication setting	1	0	☆
FA-03	PID have an effect on direction	PID direction	0: forward 1: backward	1	0	☆
FA-04	PID setting feedback range	PID range	0~65535	1	1000	☆
FA-05	proportional gain P	proportional gain P	0.0~100.0	0.1	20.0	☆
FA-06	integral time I	integral time I	0.01s~10.00s	0.01s	2.00s	☆
FA-07	differential time D	differential time D	0.000s~10.000s	0.01s	0.00s	☆
FA-08	PID reversal end frequency	reversal end frequency	0.00~Max. frequency	0.01Hz	2.00Hz	☆
FA-09	windage limit	windage limit	0.0%~100.0%	0.1%	0.0%	☆
FA-10	differential limit range	differential limit range	0%~100%	1%	5%	☆

Group FB Frequency,length and take count of

Group FB Frequency,length and take count of						
FB-00	swing setting type	swing setting type	0: relative to center frequency 1: relative to Max. frequency	0.01	0.00	☆
FB-01	frequency amplitude	frequencyamplitude	0.0%~100.0%	0.1%	0.0%	☆
FB-02	snap back frequencyamplitude	snap back frequency amplitude	0.0%~50.0%	0.1%	0.0%	☆
FB-03	frequency period	frequency period	0.1s~3000.0s	0.1s	10.0s	☆
FB-04	triangle wave rise timecoefficient	triangle wave rise time	0.1%~100.0%	0.1%	50.0%	☆
FB-05	setting length	setting length	0m~ 65535m	1m	1000m	☆
FB-06	actual length	actual length	0m~ 65535m	1m	0m	☆
FB-07	pulse/m	pulse/m	0.1~6553.5	0.1	100.0	☆
FB-08	setting numerical value	setting numerical value	1~65535	1	1000	☆
FB-09	appoint numerical value	appoint numerical value	1~65535	1	1000	☆

Group FC Multiple sections of speeds PLC

Group FC Multiple sections of speeds PLC						
FC-00	multiple sections of speeds 0	**multiple sections of speeds 0**	negative Max. frequency ~Max. frequency	0.1Hz	0.0Hz	☆
FC-01	multiple sections of speeds 1	multiple sections of speeds 1	negative Max. frequency ~Max. frequency	0.1Hz	0.0Hz	☆
FC-02	multiple sections of speeds 2	multiple sections of speeds 2	negative Max. frequency ~Max. frequency	0.1Hz	0.0Hz	☆
FC-03	multiple sections of speeds 3	multiple sections of speeds 3	negative Max. frequency ~Max. frequency	0.1Hz	0.0Hz	☆
FC-04	multiple sections of speeds 4	multiple sections of speeds 4	negative Max. frequency ~Max. frequency	0.1Hz	0.0Hz	☆
FC-05	multiple sections of speeds 5	multiple sections of speeds 5	negative Max. frequency ~Max. frequency	0.1Hz	0.0Hz	☆
FC-06	multiple sections of speeds 6	multiple sections of speeds 6	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-07	multiple sections of speeds 7	multiple sections of speeds 7	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-08	multiple sections of speeds 8	multiple sections of speeds 8	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-09	multiple sections of speeds 9	multiple sections of speeds 9	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-10	multiple sections of speeds 10	multiple sections of speeds 10	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-	multiple sections of	multiple sections of	negative Max. frequency~Max.	0.1Hz	0.0Hz	☆

11	speeds 11	speeds 11	frequency			
FC-12	multiple sections of speeds 12	multiple sections of speeds 12	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-13	multiple sections of speeds 13	multiple sections of speeds 13	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-14	multiple sections of speeds 14	multiple sections of speeds 14	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-15	multiple sections of speeds 15	multiple sections of speeds 15	negative Max. frequency~Max. frequency	0.1Hz	0.0Hz	☆
FC-16	PLC running type	PLC type	0: singly running then stop and downtime 1: when singly running finished keep end value 2: all the while circulated	1	0	☆
FC-17	when PLC power off choose memoried or not	PLC memoried	0: power off don't memoried 1: power off memoried	0	0	☆
FC-18	No. 0 runtime of PLC	No.0 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-19	choose No.0acceleration and decelerationtime of PLC	No.0 acceleration and deceleration	0~3	1	0	☆
FC-20	No.1 runtime of PLC	No.1 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-21	choose No.1acceleration and decelerationtime of PLC	No.1 acceleration and deceleration	0~3	1	0	☆
FC-22	No. 2 runtime of PLC	No.2 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-23	choose No.2acceleration and decelerationtime of PLC	No.2 acceleration and deceleration	0~3	1	0	☆
FC-24	No. 3 runtime of PLC	No. 3 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-25	choose No.3acceleration and deceleration time of PLC	No. 3 acceleration and deceleration	0~3	1	0	☆
FC-26	No. 4 runtime of PLC	No. 4 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-27	choose No.4acceleration and decelerationtime of PLC	No. 4 acceleration and deceleration	0~3	1	0	☆
FC-28	No. 5 runtime of PLC	No. 5 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
	choose	No. 5				

FC-29	No.5acceleration and decelerationtime of PLC	acceleration and deceleration	0~3	1	0	☆
FC-30	No. 6 runtime of PLC	No.6 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-31	choose No.6acceleration and decelerationtime of PLC	No.6 acceleration and deceleration	0~3	1	0	☆
FC-32	No.7 runtime of PLC	No.7 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-33	choose No.7acceleration and decelerationtime of PLC	No.7 acceleration and deceleration	0~3	1	0	☆
FC-34	No.8 runtime of PLC	No.8 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-35	choose No.8acceleration and decelerationtime of PLC	No.8 acceleration and deceleration	0~3	1	0	☆
FC-36	No. 9 runtime of PLC	No.9 runtime	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-37	choose No.9acceleration and decelerationtime of PLC	**No.9 acceleration and deceleration**	0~3	1	0	☆
FC-38	No.10 runtime of PLC	**No.10 runtime**	0.0s(h)~6553.5s(h)	0.1s (h)	0.0s (h)	☆
FC-39	choose No.10acceleration and decelerationtime of PLC	**No.10 acceleration and deceleration**	0~3	0.1s (h)	0.0s (h)	☆
FC-41	No. 11 runtime of PLC	**No.11 runtime**	**0.0s(h)~6553.5s(h)**	**0.1s(h)**	**0.0s(h)**	☆
FC-41	**choose No.11acceleration and decelerationtime of PLC**	No.11 acceleration and deceleration	**0~3**	1	0	☆
FC-42	**No. 12 runtime of PLC**	**No.12 runtime**	0.0s(h)~6553.5s(h)	**0.1s(h)**	0.0s (h)	☆
FC-43	**choose No.12acceleration and decelerationtime of PLC**	**No.12 acceleration and deceleration**	0~3	1	0	☆
FC-44	**No. 13 runtime of PLC**	**No.13 runtime**	**0.0s(h)~6553.5s(h)**	**0.1s(h)**	**0.0s(h)**	☆
FC-45	**choose No.13acceleration and decelerationtime of PLC**	**No.13 acceleration and deceleration**	0~3	1	0	☆
FC-46	**No. 14 runtime of PLC**	**No.14 runtime**	0.0s(h)~6553.5s(h)	**0.1s(h)**	**0.0s(h)**	☆
	choose No.14acceleration	**No.14**				

FC-47	**and decelerationtime of PLC**	**acceleration and deceleration**	0~3	1	0	☆
FC-48	**No. 15 runtime of PLC**	**No.15 time**	0.0s(h)~6553.5s(h)	**0.1s(h)**	**0.0s(h)**	☆
FC-49	**choose No.15acceleration and decelerationtime of PLC**	**No.15 acceleration and deceleration**	0~3	1	0	☆
FC-50	**choose PLCruntime unit**	**choose runtime unit**	**0: s(s)** **1: h(h)**	1	0	☆

Group FD Communication parameter

Group FD Communication parameter						
FD-00	baud rate	baud rate	0: 300BPS 1: 600BPS 2: 1200BPS 3: 2400BPS 4: 4800BPS 5: 9600BPS 6: 19200BPS 7: 38400BPS	1	5	☆
FD-01	data format	data format	0: none parity 1: even parity 2: odd parity	1	0	☆
FD-02	local address	local address	1~247, 0 is broadcast address	1	1	☆
FD-03	respond delay	responded delay	0ms~20ms	1	2	☆
FD-04	communication timeout	communication timeout	0.0 (invalidation),0.1s~60.0s	0.1s	0.0	☆
FD-05	choose communication protocol	choose communication protocol	0: nonstandard MODBUS protocol 1: standard MODBUS protocol	1	0	☆
Group FE function group be reserved						

Group FF Manufacturer parameter

Group FF Manufacturer parameter						
FF-00	manufacturer password	manufacturer password	reserve	reserve	reserve	*

Group FP User password and initialization

Group FP user password						
FP-00	user password	user password	0~65535	1	0	☆
FP-01	initialized parameter	initialized parameter	0: no operation 1: reset 2: clean up malfunction log	1	0	★

Appendix

The appendix includes:

Autoshop shortcut function table, system special component table, PLC error message description table, error code storage table, H2U series PLC built-in MODBUS slave-station communication protocol description, enhanced function description for H2U series high-speed processing command, and functional description for some special relay and register.

Shortcut keys

General shortcut keys

Shortcut key	Function
CTRL + N	New project
CTRL + O	Open project
CTRL + C	Copy
CTRL + X	Cut
CTRL + V	Paste
CTRL + F	Find
CTRL + H	Replace
CTRL + G	Positioning
CTRL + Z	Undo
CTRL + Y	Redo
CTRL + S	Save
CTRL + P	Print
Delete	Delete selected content
F1	Online help
F5	PLC RUN
F6	PLC STOP
F7	Compile all files
CTRL + F7	Compile current file
F8	Download
F9	Upload
F3	Monitor
F4	Write component value compulsively (in monitor mode)
SHIFT + Insert	Insert row
SHIFT + Delete	Delete row

Shortcut keys of editing ladder diagram

Shortcut key	Function
CTRL + 1	NO(normally open) contact

CTRL + 2	NC(normally closed) contact
CTRL + 3	Rising edge contact
CTRL + 4	Falling edge contact
CTRL + 5	Stpeper contact
CTRL + 6	Comparison contact
CTRL + 7	Coil
CTRL + 8	Application instruction
CTRL + 9	Subroutine call
CTRL + M	Insert network comment
CTRL + ↓	Vertical bar
CTRL + SHIFT +↓	Delete vertical bar
CTRL + →	Horizontal line
SHIFT + N	Insert network
SHIFT + D	Delete network
CTRL + E	Component comment (in ladder diagram conponent comment mode)
SHIFT + Insert	Insert row
SHIFT + Delete	Delete row
Insert	Insert col

Shortcut keys of editing sequence function diagram

Shortcut key	Function
CTRL + 1	Ladder diagram block
CTRL + 2	Initial stepper symbol
CTRL + 3	General stepper symbol
CTRL + 4	Transfer symbol
CTRL + 5	Jump symbol
CTRL + 6	Loop symbol
CTRL + 7	Selection Branch
CTRL + 8	Selection Merge
CTRL + 9	Simultaneous branch
CTRL + 0	Simultaneous merge
CTRL + ↓	Vertical bar
CTRL + →	Horizontal line
SHIFT + Insert	Insert row
SHIFT + Delete	Delete row
CTRL + L	Built-in ladder diagram and return

Special system component

M8000~M8255£¬D8000~D8255 are defined as special components of which functions are described as following:

M component	**M component description**	**D component**	**D component description**
System operation status			
M8000	ON when user program is running	D8000	Monitor timer of user program operation

M8001	M8000 status is inverted	D8001	Single board program version, for example 24100 H2U = 24, 100 version V1.00
M8002	ON during first period of user program	D8002	Program capacity, 4K, 8K and 16K etc.
M8003	M8002 status is inverted	D8003	Fixed value 0X10, internal memory of PLC
M8004	If any of M8060~M8067 [except for M8062] is ON then M8004 is effective	D8004	Wrong BCD value of M8060~M8067, normally 0
M8005	Actuated when battery voltage is too low	D8005	BCD current value of battery voltage
M8006	Actuated when battery is too low [latch]	D8006	Threshold value of low battery voltage detection, initial value is 2.6V
M8007	AC power lost for 5ms then M8007&M8008 will be actuated, but the program continue running within D8008	D8007	Time of saving M8007 actions, reset to 0 when power is lost
M8008	If the power is lost within D8008, then the user program will stop running when M8008 changes from ON¡úOFF. M8000 is OFF	D8008	AC power lost detection time, default 10ms
M8009	Actuated when the extension unit loses 24V power	D8009	Module number of extension unit which loses 24V power
System clock			
M8010	reserved	D8010	Current scan time, from step 0 of user program (0.1 ms)
M8011	Clock oscillator of which period is 10ms	D8011	Minimum scan time (0.1 ms)
M8012	Clock oscillator of which period is 100ms	D8012	Maximum scan time (0.1ms)
M8013	Clock oscillator of which period is 1s	D8013	Clock second (0~59)
M8014	Clock oscillator of which period is 1 minute	D8014	Minute of Real-time clock (0~59)
M8015	Clock stop and preset	D8015	Hour of Real-time clock (0~23)
M8016	Stop clock read and display	D8016	Day of Real-time clock (1~31)
M8017	¡À30 seconds correction	D8017	Month of Real-time clock(1~12)
M8018	Installation detection	D8018	Year of Real-time clock (2000~2099)
M8019	Real-time clock (RTC) error	D8019	Week of real-time clock
Instruction flags			
M8020	Operation Zero flag	D8020	Input filter constant 0~60 of X000~X007 [default 10ms]
M8021	Operation Borrow flag	D8021	Reserved
M8022	Operation Carry flag	D8022	Reserved
M8023	Reserved	D8023	Reserved
M8024	BMOV instruction direction	D8024	Reserved
M8025	HSC instruction mode	D8025	Reserved
M8026	RAMP instruction mode	D8026	Reserved

M8027	PR mode	D8027	Reserved
M8028	Reserved	D8028	The same address with Z0
M8029	Instruction (PLSR and so on) execution complete	D8029	The same address with V0
PLC operation mode			
M8030	If it¡¯s ON then even battery voltage is low the alarm BATT.V LED not lit	D8030	Reserved
M8031	Non-latch memory all clear when it's ON	D8031	Reserved
M8032	Latch memory all clear when it's ON	D8032	Reserved
M8033	When ON memory hold in "stop" mode	D8033	Reserved
M8034	When ON all PLC output is OFF state	D8034	Reserved
M8035	Forced operation command 1	D8035	Reserved
M8036	Force operation command 2	D8036	Reserved
M8037	Force stop command	D8037	Reserved
M8038	Communication setup flags	D8038	Reserved
M8039	Constant scan mode	D8039	Constant scan time,default 0, the unit is ms
Step ladder flag			
M8040	STL transfer disable	D8040	Up to 8 active STL states, from the range S0 to S899, are stored in D8040 to D8047 in ascending numerical order.
M8041	Transfer start	D8041	
M8042	A pulse output is given in response to a start input	D8042	
M8043	ON during the last state of ZERO RETURN mode	D8043	
M8044	ON when the machine zero is detected	D8044	
M8045	Disables the "all output reset" function when the operation mode is changed	D8045	
M8046	ON when STL monitoring has been enabled (M8047) and there is an active STL state	D8046	
M8047	When ON D8040 to D8047 are enabled for active STL step monitoring	D8047	
M8048	When M8049 is ON, anyone from S900~S999 is enabled.	D8048	Reserved
M8049	When ON D8049 is enabled for active annunciator state monitoring	D8049	save S900~S999's alarm min. address No.
Interrupt control flags			
M8050	Drive I00¡õ interrupt disabled	D8050	Reserved
M8051	Drive I10¡õ interrupt disabled	D8051	Reserved
M8052	Drive I20¡õ interrupt disabled	D8052	Reserved
M8053	Drive I30¡õ interrupt disabled	D8053	Reserved
M8054	Drive I40¡õ interrupt disabled	D8054	Reserved

M8055	Drive I50¡õ interrupt disabled			D8055	Reserved
M8056	Drive I6¡õ¡õ interrupt disabled			D8056	Reserved
M8057	Drive I7¡õ¡õ interrupt disabled			D8057	Reserved
M8058	Drive I8¡õ¡õ interrupt disabled			D8058	Reserved
M8059	Drive counter interrupt disabled			D8059	Reserved
Error detection devices					
components	name	Program error LED	PLC status		
M8060	I/O configuration error []	OFF	RUN	D8060	The first I/O number of the unit or block causing the error
M8061	PLC hardware error	Flash	STOP	D8061	Error code for hardware error
M8062	PLC communication error	OFF	RUN	D8062	Error code for Communications error
M8063	Parallel link/ general communication error	OFF	RUN	D8063	Error code for parallel link error
M8064	Parameter error	Flash	STOP	D8064	Error code identifying parameter error
M8065	Syntax error	Flash	STOP	D8065	Error code identifying syntax error
M8066	Program error	Flash	STOP	D8066	Error code identifying program construction error
M8067	Operation error	OFF	RUN	D8067	Error code identifying operation error
M8068	Operation error latch	OFF	RUN	D8068	Operation error step number latched
M8069	Reserved			D8069	Step numbers for found errors corresponding to flags M8065 to M8067
Link function					
M8070	Driven when the PLC is a master station in a parallel link application			D8070	Parallel link watchdog time - 500 ms
M8071	Driven when the PLC is a slave station in a parallel link application			D8071	Reserved
M8072	ON while the PLC is operating in a parallel link			D8072	Reserved
M8073	ON when M8070/ M8071 are incorrectly set during parallel link operations			D8073	Reserved
Tracking sampling					
M8074	Reserved			D8074	Remain number of tracking sampling
	Tracking Sampling get ready to				Tracking sampling No. setup

M8075	begin instruction	D8075	(1~512)
M8076	Tracking sampling complete,then instruction execution start	D8076	Tracking sampling cycle
M8077	Tracking sampling while execution monitoring	D8077	Trigger Designation
M8078	Tracking sampling when execution complete monitoring	D8078	Components address number setup of trigger condition
M8079	Sampling data tracking more than D8075	D8079	Tracking sampling data pointer
M8080	Reserved	D8080	Bit component address number No.0
M8081	Reserved	D8081	Bit component address number No.1
M8082	Reserved	D8082	Reserved
M8083	Reserved	D8083	Reserved
M8084	High speed counter multiple interrupt enabled (default OFF)	D8084	Counter sequence number of high speed counter multiple interrupts
M8085	Output initialization flag of Y0 port	D8085	Default data of multiple interrupts are 0
M8086	Output initialization flag of Y1 port	D8086	Corresponding D component sequence number
M8087	Output initialization flag of Y2 port	D8087	Reserved
M8088	Output initialization flag of Y3 port	D8088	Reserved
M8089	Output initialization flag of Y4 port	D8089	Reserved
M8090	Y0 Output complete interrupt enabled	D8090	Reserved
M8091	Y1 Output complete interrupt enabled	D8091	Reserved
M8092	Y2 Output complete interrupt enabled	D8092	Reserved
M8093	Y3 Output complete interrupt enabled	D8093	Reserved
M8094	Y4 Output complete interrupt enabled	D8094	Reserved
M8095	Reserved	D8095	Reserved
M8096	Reserved	D8096	Word component address number No.0
M8097	Reserved	D8097	Word component address number No.1
M8098	Reserved	D8098	Word component address number No.2
High speed ring counter			
M8099	High speed ring counter operation	D8099	[0 to 32767] increased action ring-counter (0.1 ms)
Miscellaneous Devices			
M8100	SPD(X000)- pulse numbers/minute	D8100	Reserved

M8101	SPD(X001)- pulse numbers/minute	D8101	Single board program version, for example 24100 H2U = 24, 100 version V1.00
M8102	SPD(X002)- pulse numbers/minute	D8102	Program capacity provided by system to user program
M8103	SPD(X003)- pulse numbers/minute	D8103	Reserved
M8104	SPD(X004)- pulse numbers/minute	D8104	Acceleration time when executing DRVI and DRVA [default 100], M8135 determines that it¡¯s whether effective or not [Y0]
M8105	SPD(X005)- pulse numbers/minute	D8105	Acceleration time when executing DRVI and DRVA [default 100], M8135 determines that it¡¯s whether effective or not [Y1]
M8106	Reserved	D8106	Acceleration time when executing DRVI and DRVA [default 100], M8135 determines that it¡¯s whether effective or not [Y2]
M8107	Reserved	D8107	Acceleration time when executing DRVI and DRVA [default 100], M8135 determines that it¡¯s whether effective or not [Y3]
M8108	Reserved	D8108	Acceleration time when executing DRVI and DRVA [default 100], M8135 determines that it¡¯s whether effective or not [Y4]
M8109	Output refresh error	D8109	Output refresh error address number
COM0 communication and link			
M8110	Reserved	D8110	Communication format, the interface configuration with a default of 0
M8111	Sending and waiting (RS instruction)	D8111	Station number settings, the interface configuration settings with a default of 1
M8112	Sending flag (RS instruction) Instruction execution status (MODBUS)	D8112	Amount of remaining data to be transmitted (Only to RS instruction)
M8113	Receiving complete flag (RS) Communication error flag (MODBUS)	D8113	Amount of data already received (Only to RS instruction)
M8114	Receiving (only to RS instruction)	D8114	Start character STX (Only to RS instruction)
M8115	Reserved	D8115	Termination character ETX (Only to RS instruction)

M8116	Reserved	D8116	Communication protocol, the interface configuration with a default of 0
M8117	Reserved	D8117	Computer link protocol of data starting address
M8118	Reserved	D8118	Computer link protocol sending data amount
M8119	timeout judgement	D8119	Communication overtime judgement,? the interface configuration settings with a default of 10£¨100ms£©
COM1 communication link			
M8120	Reserved	D8120	Communication format, the interface configuration with a default of 0
M8121	Sending and waiting (RS instruction)	D8121	Station number settings, the interface configuration settings with a default of 1
M8122	Sending flag (RS instruction) Instruction execution status (MODBUS)	D8122	Amount of remaining data to be transmitted (Only to RS instruction)
M8123	Receiving complete flag (RS) Communication error flag (MODBUS)	D8123	Amount of data already received (Only to RS instruction)
M8124	Receiving (only to RS instruction)	D8124	Start character STX (Only to RS instruction)
M8125	Reserved	D8125	Termination character ETX (Only to RS instruction)
M8126	Reserved	D8126	Communication protocol, the interface configuration with a default of 0
M8127	Reserved	D8127	Computer link protocol of data starting address
M8128	Reserved	D8128	Computer link protocol sending data amount
M8129	timeout judgement	D8129	Communication overtime judgement,? the interface configuration settings with a default of 10£¨100ms£©
High speed & positioning			
M8130	Control mode of HSZ instruction platform	D8130	Special bit for high-speed model (record number)
M8131	Paralleled with M8130	D8131	HSZ & PLSY completion mark of comparison mode (record number)
M8132	HSZ&PLSY speed mode	D8132	HSZ & PLSY frequency control mode
M8133	Paralleled with M8132	D8133	
M8134	Reserved	D8134	
	Y0 speed-down time and pulse		

M8135	output can be change to be enabled [ON-PLSR,DRVI,DRVA]	D8135	Completion mark for HSZ & PLSY frequency control mod
M8136	Y1 speed-down time and pulse output can be change to be enabled[ON-PLSR,DRVI,DRVA]	D8136	The total number of Y000&Y001 output pulses
M8137	Y2 speed-down time and pulse output can be change to be enabled[ON-PLSR,DRVI,DRVA]	D8137	
M8138	Y3 speed-down time and pulse output can be change to be enabled[ON-PLSR,DRVI,DRVA]	D8138	Reserved
M8139	Y4 speed-down time and pulse output can be change to be enabled[ON-PLSR,DRVI,DRVA]	D8139	Reserved
M8140	CLR signal output function of ZRN is enabled.	D8140	PLSY&PLSR output Y000 corresponding cumulative value for the pulse number
M8141	Reserved	D8141	
M8142	Reserved	D8142	PLSY&PLSR output Y001 corresponding cumulative value for the pulse number
M8143	Reserved	D8143	
M8144	Reserved	D8144	
M8145	Y000 pulse output stop	D8145	The offset speed when DRVI,DRVA execution
M8146	Y001 pulse output stop	D8146	Maximum speed of DRVI,DRVA execution[Default 100,000]
M8147	Y000 pulse output monitor	D8147	
M8148	Y001 pulse output monitor	D8148	acceleration and deceleration time when DRVI,DRVA execution[Default 100]
M8149	Y002 pulse output monitor	D8149	Reserved
M8150	Y003 pulse output monitor	D8150	PLSY&PLSR output Y002 corresponding cumulative value for the pulse number
M8151	Y004 pulse output monitor	D8151	
M8152	Y002 pulse output monitor	D8152	PLSY&PLSR output Y003 corresponding cumulative value for the pulse number
M8153	Y003 pulse output stop	D8153	
M8154	Y004 pulse output stop	D8154	PLSY&PLSR output Y004 corresponding cumulative value for the pulse number
M8155	Reserved	D8155	
M8156	Reserved	D8156	Clear definition of Y0 port signal (ZRN)[Default 5=Y005]
M8157	Reserved	D8157	Clear definition of Y1 port signal (ZRN)[Default 6=Y006]
Extension function			

M8158	Reserved	D8158	Clear definition of Y2 port signal (ZRN)[Default 7=Y007]
M8159	Reserved	D8159	Clear definition of Y3 port signal (ZRN)[Default 8=Y010]
M8160	Selection of XCH operation to swap bytes in a single data word	D8160	Clear definition of Y4 port signal (ZRN)[Default 9=Y011]
M8161	Selection of 8 bit operations for applied instructions ASC, RS, ASCI, HEX, CCD	D8161	Reserved
M8162	High speed mode for parallel connection	D8162	Reserved
M8163	Reserved	D8163	Reserved
M8164	(FROM/TO) Move points variable mode	D8164	(FROM/TO) Move points fixed mode
M8165	Reserved	D8165	When the PLSR, DRVI, DR VA are in execution, the deceleration time [default 100] is determined by M8135 whether it is enabled. [Y0]
M8166	Reserved	D8166	When the PLSR, DRVI, DR VA are in execution, the deceleration time [default 100] is determined by M8136 whether it is enabled. [Y1]
M8167	(HEY)HEX data processing function	D8167	When the PLSR, DRVI, DR VA are in execution, the deceleration time [default 100] is determined by M8137 whether it is enabled. [Y2]
M8168	(SMOV)HEX data processing function	D8168	When the PLSR, DRVI, DR VA are in execution, the deceleration time [default 100] is determined by M8138 whether it is enabled. [Y3]
M8169	Reserved	D8169	When the PLSR, DRVI, DR VA are in execution, the deceleration time [default 100] is determined by M8139 whether it is enabled. [Y4]
Pulse capture		Communication link	
M8170	X000 pulse capture	D8170	Reserved
M8171	X001 pulse capture	D8171	Reserved
M8172	X002 pulse capture	D8172	Reserved
M8173	X003 pulse capture	D8173	Station No. set?status
M8174	X004 pulse capture	D8174	Communication sub-station set status
M8175	X005 pulse capture	D8175	Refresh range set status
M8176	Reserved	D8176	Station No. setting
M8177	Reserved	D8177	Communication sub-station number setting
M8178	Reserved	D8178	Refresh range setting
M8179	Reserved	D8179	Retry count setting

M8180	Reserved	D8180	Communication overtime setup
Communication link		Index addressing	
M8181	Reserved	D8181	Reserved
M8182	Reserved	D8182	Bit component address number No.2/Z1 register contents
M8183	Data transfer master station error	D8183	Bit component address number No.3/V1 register contents
M8184	Data transfer slave station 1 error	D8184	Bit component address number No.4/Z2 register contents
M8185	Data transfer slave station 2 error	D8185	Bit component address number No.5/V2 register contents
M8186	Data transfer slave station 3 error	D8186	Bit component address number No.6/Z3 register contents
M8187	Data transfer slave station 4 error	D8187	Bit component address number No.7/V3 register contents
M8188	Data transfer slave station 5 error	D8188	Bit component address number No.8/Z4 register contents
M8189	Data transfer slave station 6 error	D8189	Bit component address number No.9/V4 register contents
M8190	Data transfer slave station 7 error	D8190	Bit component address number No.10/Z5 register contents
M8191	Data transferring	D8191	Bit component address number No.11/V5 register contents
M8192	Reserved	D8192	Bit component address number No.12/Z6 register contents
M8193	Reserved	D8193	Bit component address number No.13/V6 register contents
M8194	Reserved	D8194	Bit component address number No.14/Z7 register content
M8195	C251 Double-frequency	D8195	Bit component address number No.15/V7 register content
M8196	C252 Double-frequency	D8196	Reserved
M8197	C253 Double-frequency	D8197	Reserved
M8198	C254 Double-frequency	D8198	Reserved
M8199	C255 Double-frequency	D8199	Reserved
Up/down counter control and status		Communication link	
M8200	C200 control	D8200	Reserved
M8201	C201 control	D8201	Currently connection scan time
M8202	C202 control	D8202	Maximum connection scan time
M8203	C203 control	D8203	Master station communication error number
M8204	C204 control	D8204	Slave station 1 communication error number
M8205	C205 control	D8205	Slave station 2 communication error number
M8206	C206 control	D8206	Slave station 3 communication error number
M8207	C207 control	D8207	Slave station 4 communication error number
M8208	C208 control	D8208	Slave station 5 communication error number

M8209	C209 control	D8209	Slave station 6 communication error number
M8210	C210 control	D8210	Slave station 7 communication error number
M8211	C211 control	D8211	Master station communication error code
M8212	C212 control	D8212	Slave station 1 communication error code
M8213	C213 control	D8213	Slave station 2 communication error code
M8214	C214 control	D8214	Slave station 3 communication error code
M8215	C215 control	D8215	Slave station 4 communication error code
M8216	C216 control	D8216	Slave station 5 communication error code
M8217	C217 control	D8217	Slave station 6 communication error code
M8218	C218 control	D8218	Slave station 7 communication error code
M8219	C219 control	D8219	Reserved
M8220	C220 control	D8220	Reserved
M8221	C221 control	D8221	Reserved
M8222	C222 control	D8222	Reserved
M8223	C223 control	D8223	Reserved
M8224	C224 control	D8224	Reserved
M8225	C225 control	D8225	Reserved
M8226	C226 control	D8226	Reserved
M8227	C227 control	D8227	Reserved
M8228	C228 control	D8228	Reserved
M8229	C229 control	D8229	Reserved
M8230	C230 control	D8230	Reserved
M8231	C231 control	D8231	Reserved
M8232	C232 control	D8232	Reserved
M8233	C233 control	D8233	Reserved
M8234	C234 control	D8234	Reserved
M8235	C235 control	D8235	Reserved
M8236	C236 control	D8236	Reserved
M8237	C237 control	D8237	Reserved
M8238	C238 control	D8238	Reserved
M8239	C239 control	D8239	Reserved
M8240	C240 control	D8240	Reserved
M8241	C241 control	D8241	Reserved
M8242	C242 control	D8242	Reserved
M8243	C243 control	D8243	Reserved
M8244	C244 control	D8244	Reserved
M8245	C245 control	D8245	Reserved
M8246	C246 control	D8246	Reserved
M8247	C247 control	D8247	Reserved
M8248	C248 control	D8248	Reserved

M8249	C249 control	D8249	Reserved
M8250	C250 control	D8250	Reserved
M8251	C251 control	D8251	Reserved
M8252	C252 control	D8252	Reserved
M8253	C253 control	D8253	Reserved
M8254	C254 control	D8254	Reserved
M8255	C255 control	D8255	Reserved

Error code storage

The errors of H2U will be checked on time, and the error code will be stored into special data registers D8060~D8067.

Error item	Power OFF→ON	The first time STOP→RUN after power ON	Others
M8060 I/O address error	Check up	Check up	In operation
M8061 PC hardware error	Check up	-	In operation
M8062 PC/PP communication error	-	-	Receiving signal from PP
M8063 connection module communication error	-	-	Receiving signal from another side
M8064 parameter error M8065 syntax error M8066 circuit error	Check up	Check up	Program changing (STOP) Program transferring (STOP)
M8087 operation error M8088 operation error latch	-	-	In operation (RUN)

Each of D8060~D8067 stores one error. If the same error item generates errors more than once then the current error code is still stored when eliminating the error causes. If there is no error then "0" will be stored.

Error messages

Following is the error code and messages that will be stored in the special data registers D8060～D8067:

Type	Error code	Causes	Trouble-shooting
I/O structural error M8060(D8060)	Example 1020	I/O start-up component number "1020" is not installed:	The input and output relay numbers will be written into the program. Programmable controller can continue the

Continue operation		1=output X (0=output Y), 020 = component number	operation. Programmer please modify the program.
PC hardware error M8061(D8061) Stop operation	0000	Normal	Check extension cable's connection.
	6101	RAM error	
	6102	Calculation circuit error	
	6103	I/O bus error (when M8069 is driven)	
	6104	Expansion devices, below 24V (when M8069 is ON)	
	6105	Monitoring timer error	If the calculation time exceeds the value of D8000, please check the programming.
PC/PP communication error M8062(D8062) Continue operation	0000	Normal	Check the connections between the programmable controller and the program panel (PP) or program interface
	6201	Odd/even parity error overtime error framing error	
	6202	Communication character error	
	6203	Checksum of communication data differs	
	6204	Data format error	
	6205	Instruction error	
	0000	No abnormity	Check to ensure that the power of both programmable controllers is ON. In addition, check to ensure that the connections between the adapter and the controller and between adapters are correct.
	6301	Odd/even parity error Overtime error Framing error	
	6302	Communication character error	
	6303	Communication data checksum differs	
	6304	Data format error	
	6305	Instruction error	
	6306	Monitor timer overflow	
	6307~ 6311	None	
	6312	Parallel link character error	
	6313	Parallel link checksum error	
	6314	Parallel link format error	
	6330	MODBUS slave address setup error	In the event of a COM0

Parallel link communication error M8063 (D8063).Continue running.	6331	Data frame length error	communication error, please check to ensure that the COM0 communication cable is connected correctly. Check to ensure that the communication format of both sides matches each other. Check to ensure that the communication protocols match. Check to ensure that the system is powered on because COM0 can only be used as a free port in the power-on state. Otherwise, it can only be used as download port. Check to ensure that the JP0 jumper is inserted because COM0 can only be used as a RS485 free port when the jumper is open. If the JP0 is closed, then COM0 can only be used as a monitor or a download port and is in RS422 mode.
	6332	Address error	
	6333	CRC check error	
	6334	Function cede not supported	
	6335	Receiving overtime	
	6336	Data error	
	6337	Buffer overflow	
	6338	Frame error	
	6340	MODBUS slave address setup error	In the event of a COM1 communication error, please
	6341	Data frame length error	
	6342	Address error	
	6343	CRC check error	
	6344	Function code not supported	
	6345	Receiving overtime	
	6346	Data error	
	6347	Buffer overflow	
	6348	Frame error	
Parameter error M8064(D8064) Stop operation	0000	Normal	Terminate the operation of the programmable controller and set up the correct alue by using parameters.
	6401	Inconsistant checksum of programs	
	6402	Storage capacity setting error	
	6403	Storage setting error	
	6404	Instruction setting error	
	6405	File register setting error	
	0000	Normal	
	6503	1. no setting value after OUT T,OUT C 2. insufficient operands of	

Syntax error M8065(D8065)		application instructions	Check the programming of each instruction. Modify the programming when errors happened.
	6504	1. duplicated labels	
	6505	Component number overflow	
	6506	Undefined instruction	
	6507	Incorrect volume label (P) definition	
	6508	Incorrect interrupt input(I) definition	
	6509	Duplicated interrupt input and high speed counter input	
Circuit error M8066(D8066)	0000	Normal	Incorrect instruction group or instruction relationship can cause errors. It's required to change the relationship of instructions in program to make corrections.
	6605	1. MPS is used continuously for more than 9 times 2. MC, MCR, I (interrupt) and SRET exist under STL instruction 3. RET exists outside the STL	
	6606	1. no P (pointer) and I (interrupt) 2. No SRET and IRET 3. I(interrupt), SRET and IRET exist in main program. 4. STL, RET, MC and MCR exist in sub programs and interrupt programs	
	6607	1. Error relationship of FOR and NEXT, nesting level is more than 6 2. STL,RET,MC,MCR,IRET,SRET,FEND and END exist between FOR and NEXT	
	6608	1. Error relationship of MC and MCR 2. MCR doesn't have NO 3. SRET, IRET and I (interrupt) exist between MC and MCR	
	6618	Instructions can only be used inside the main program used outside of main program (interrupt and sub programs).	
	6619	Instructions such as STL, RET, MC, MCR, I and IRET which can not be used between FOR and	

Stop operation		NEXT are used there.	
	6620	Nesting overflow between FOR~NEXT	
	6621	Error relationship of FOR~NEXT numbers	
	6622	No NEXT instruction	
	6623	No MC instruction	
	6624	No MCR instruction	
	6625	STL is used continuously for more than 9 times	
	6626	Instructions such as MC, MCR, I, SRET and IRET cannot be used between STL as RET are used in there.	
	6627	No RET instruction	
	6628	Instructions such as I, SRET and IRET which cannot be used in main program.	
	6629	No P and I	
	6630	No SRET and IRET instruction	
	6631	SRET exists in incorrect location	
	6632	FEND exists in incorrect location	
	6635	Hardware terminals used by high speed input and output have exceeded the limit	
	0000	Normal	Check correctness of the operation, the programming, and the operands. Because they may cause error even if the syntax and circuit are correct. (example) when Z=100 and T=300 the component number will overflow even T200Z is
	6701	1. CJ and CALL don't have target address 2. Volume label exists after END instruction 3. Individual labels exist between FOR and NEXT or between subprograms	
	6702	Nesting level of CALL is more than 6	
	6704	Nesting level of FOR-NEXT is more than 6	
	6705	Operands of application instruction is outside the target component	
	6706	Component number and application data operands instruction overflow	
	6707	Accessing file registers without setting the parameters.	

	6708	FROM~TO Instruction error	not wrong.	
	6709	Others (IRET and SRET are forgot, and error relationship of FOR~NEXT ,etc.)		
	6730	Sample time (TS) is out of range (TS = 0)		
	6732	Input filter constant (a) is out of range (a<0 or 100<a)		
	6733	Proportional coefficient (KP) is out of range (KP<0)		
	6734	Integration time (TI) is out of range (TI <0)	Stop PID operation	
	6735	Differential coefficient (KD) is out of range (KD <0 or 201 <KD)		
	6736	Differential time is out of range (TD <0)		
	6740	Sample time (TS) <operation period		The setting value for generating control parameters and PID operation result are incorrect. Please check the parameters.
Operation error M8067 (D8067) Continue Operation	6742	Measurement variable overflow (△PV<32768 or <△PV)		
	6743	Deviation overflow (EV<-32768 or 32767<EV>		
	6744	Integral calculation value overflow(out of -32768～32767 range)		
	6745	Differential calculation value overflows because of KP overflow	Continue operation using the calculated data as the MAX value	
	6746	Differential calculation value overflow(out of -32768~32767 range)		
	6747	PID operation result overflow (out of -32768~32767 range)		
	6760	Number of high speed instructions (such as DHSZ and so on) exceeds the limit of 6 lines		

Function description of some special relays and registers

System running state

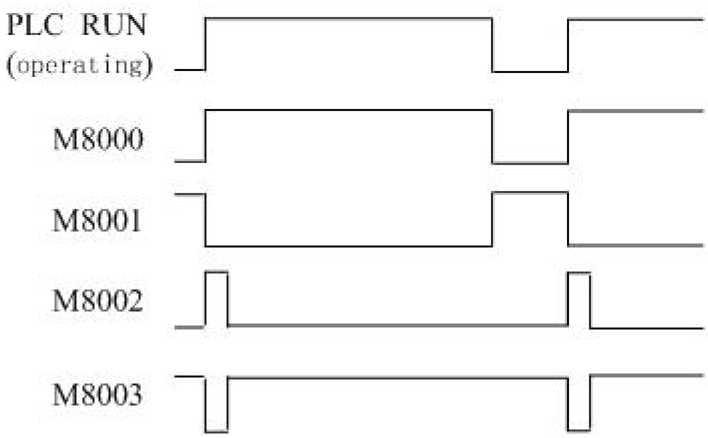
PLC running flag M8000~M8003

1. M8000: M8000 is a normally ON contact of RUN, which means it's a running monitor normally open contact (A contact). When the PLC is in RUN state the M8000 will always remain ON.

2. M8001: M8001 is a normally OFF contact of RUN, which means it's a running monitor normally closed contact (B contact). When the PLC is in RUN state the M8001 will always remain OFF.

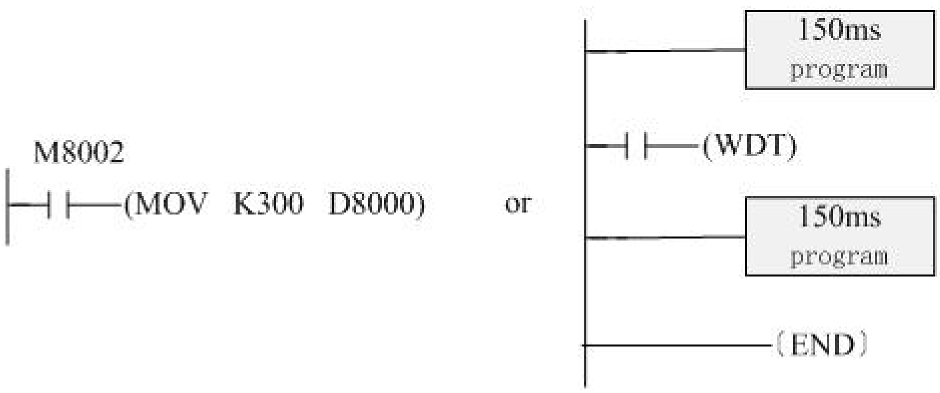
3. M8002: When the PLC starts the first scan of RUN the M8002 is ON, and then it remains in the OFF state. The width of this pulse is the time of one scan process, and this contact could be used when any initialization is needed.

4. M8003: When the PLC starts the first scan of RUN the M8003 is OFF, and then it remains ON state. So it is a negative startup pulse (instantaneous 'OFF' of RUN).



Monitor Timer D8000

1)The monitor timer is designed for monitoring the PLC scan time. If scan time exceeds the setting of the monitor timer then the ERROR red indicator will be normally ON and all the output will become Off.The initial time setting of the monitor timer is 200ms, and too complex instruction operation or too many special modules connected to PLC master would cause too much longer scan time.D8010~D8012 can be monitored so as to know whether the setting time of D8000 is exceeded. In this situation, MOV instruction could be used in the program to change the time setting of monitor timer to 300ms, or the WDT instruction could be added to the program to clear the internal monitor timer to zero when the WDT instruction is executed, so the scan time will not exceed the setting time of monitor timer.



2)The monitor timer can be set to maximum 32767ms, but please notice that the detection time of abnormal operation will be slowed down if the setting time is too long. So except for complex operations which force the scan time to be longer than 200ms, the setting time should be less than 200ms.

Single board program version D8001

Single board program version, for example D8001=24115 means: 24 means H2U series PLC, 115 means version V1.15, in other words this is a H2U type PLC whose version is V1.15.

Program capacity D8002

Program capacity, 4K,8K,16K,24K,etc. The program capacity of H2U type PLC is 24K.

Syntax check signal M8004, D8004

M8004: When any of M8060~M8067 (except for M8062) is ON, the M8004 is ON. It can be used to monitor system error of PLC. D8004: BCD value of M8060~M8067, initial value is 0.

Battery voltage measurementM8005~M8009 D8005~D8009 M8030

1)If battery voltage D8005 is lower than D8006 (initial value is 2.6V), then M8005 will output;

2)If there is any low battery voltage alarm (M8005 is ON), then the M8006 will be set to ON (latched). Program can't reset it even the PLC is restarted and reset.

3)If the system loses AC power for 5ms then the M8007 and M8008 will be actuated. PLC program will continue running if the power failure time is within D8008, or the user program will not be executed and M8000 is OFF;

4)M8009 is ON when the expanded module loses 24V power, and D8009 will record the module number;

5)If M8030 is ON then the low battery voltage alarm will be masked.

System clock

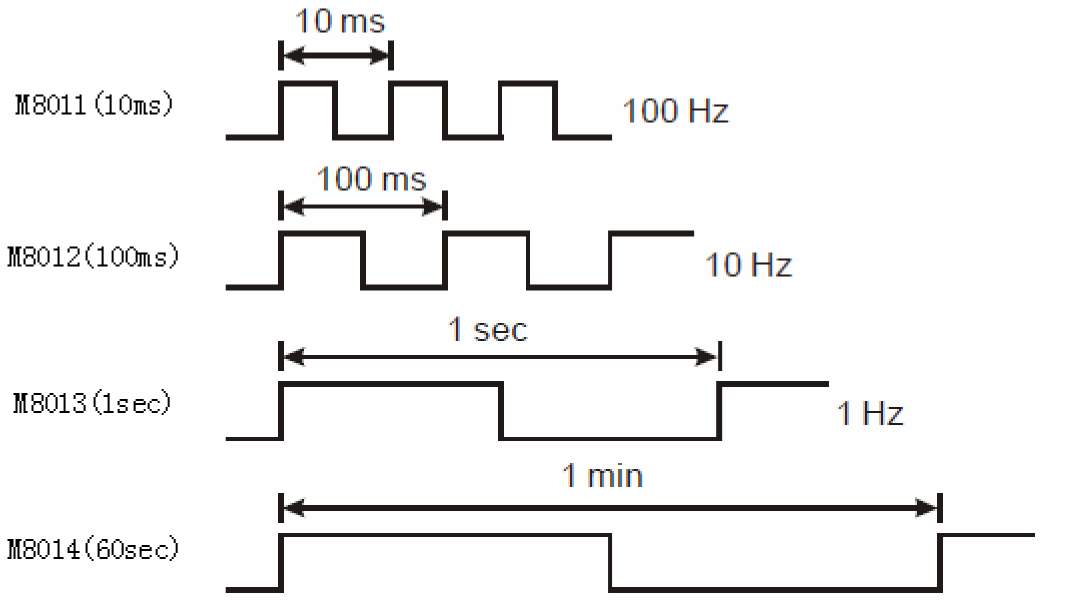
Scan time monitor D8010~D8012

The current value, minimum value and maximum value of program scan time are stored in D8010~D8012.

1. D8010: current value of scan time;
2. D8011: minimum value of scan time;
3. D8012: maximum value of scan time.

Periodical Clock crystal oscillator M8011~M8014

There are 4 types of clock crystal oscillator in the PLC. The oscillator starts automatically as soon as the PLC is powered on, and they may continue running even when PLC is in STOP state. So the startup timing of clock oscillator and RUN is not synchronous.

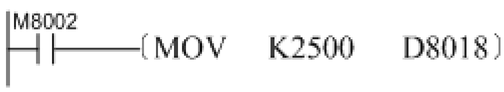


Real-time clock D8013~D8019 M8015~M8019

1.The clock stops when M8015 is ON;

2.Each time the M8017 turns ON the PLC internal clock will make a one-time ±30 second correction. In other words, if the value of D8013 is between "1" and "29", the seconds value of the internal clock will be set to "0" (and the minutes value will remain the same). On the other hand, if the value of D8013 is between "30" and "59", the seconds value will be set to "0" and the minutes value will be incremented by 1.

3.The year value is typically displayed using 2 digits (for example, the year 2009 is displayed as "09"). If a four-digit year display is desired, execute the following instruction:



Please run this program every time the PLC runs. Switching the K2000 to a four-digit display does not affect the current time value.

Address no.	Name	Action function
M8015	Clock stop and time calibration	ON when clock stop. The edge of ON→OFF is used to write time to reactivate.
M8016	Stop time display	ON when stopping clock display (timing hold active)
M8017	±30 s correction	The edge of ON→OFF is applied for amending second.(When second is of 0~29, second is set to 0.When second is 30~59, minute carries and second is set to 0.
M8018	Installation detection	Normal ON
M8019	RTC error	ON when data in special data register exceeds the setting range during time calibration.

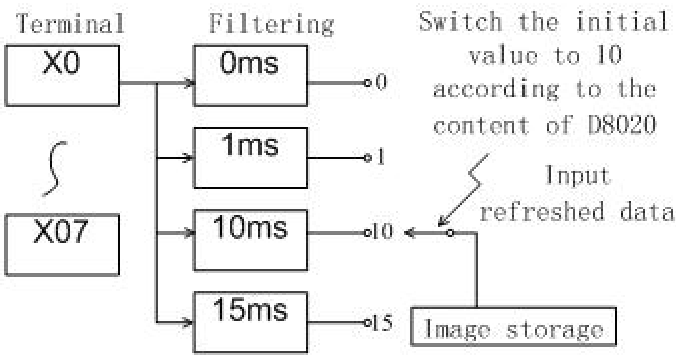
Address no.	Name	Setting range	Action function
D8013	Second	0~59	To write initial value for calibration time or read current time.
D8014	Minute	0~59	
D8015	Hour	0~23	
D8016	Day	0~31	
D8017	Month	0~12	
D8018	Year	0~99 (last two digit)	
D8019	Week	0~6 (corresponding to day"6")	

Instruction Flags

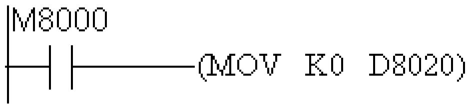
Input Filter Adjustment D8020

1.Inputs X0¨CX7. The input pulse response time can be set in D8020 a value between 0 ms and 60 ms. The default value is 10ms.

2.If high-speed counter and interrupt-insertion functions are used in the program, the filter time of the relevant input port will automatically be the shortest time, and the filter times of the remaining ports X0-CX7 will remain those of the original D8020 setting.



4.Executing the following program can change the filter constant to 0 ms, but this input port actually has a built-in RC hardware filter, so even if the constant is set to 0, the actual value will be at least 10 ms (ports X2-CX5 of 40 or 60 PLC points will have a minimum value of 50 ms).



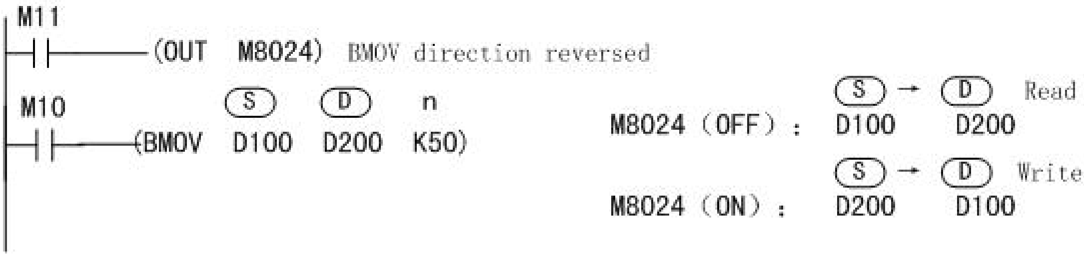
Operation Flags M8020-CM8022

The operation flags are:

1.Zero flag: M8020=ON if the result is 0
2.Borrow flag: M8021= ON if the result is less than the minimum representable value.
3.Carry flag: M8022= ON if the result exceeds the maximum representable value.

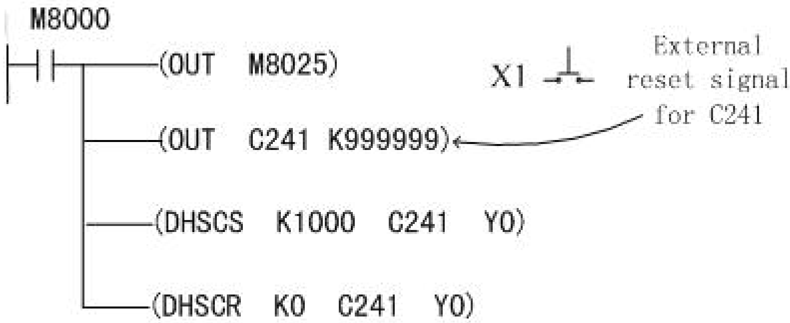
Reverse Data Move M8024

One program instruction can be used for bidirectional data transfer by controlling the reverse flag (M8024) of the BMOV instruction.



HSC Instruction Mode M8025

All the comparing output of high speed counter output contact, FNC53 (D HSCS), FNC54 (D HSCR) and FNC55 (D HSZ) instructions will actuate as the current value register of counter input changes. Even if we can change the current value (C235~C255) through data move instruction DMOV, the comparing output will not change as long as there is no counter input. Just like the description above in "Notes" section, the high speed counter C241 has a external reset terminal (R) which can be used to reset the rising edge of input signal so as to execute instruction and output the comparing result. Following is the details. External reset terminal of external reset mode C241.



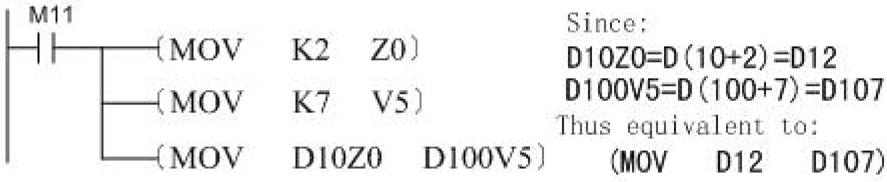
In the above example we can see that when M8025 is ON and the current value of C241 is assumed to be 2000, then the Y0 is ON. If the external reset button X1 is pressed then the current value of C241 changes to 0, and even X0 hasn't counter input the Y0 will reset anyway.

Execution finished flag M8029

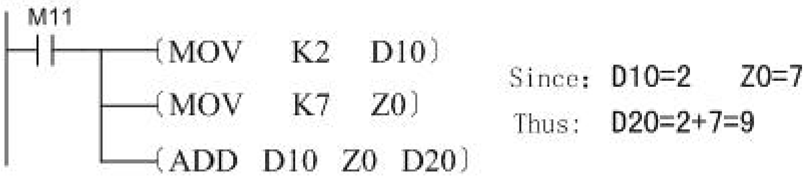
The execution finished flag of high speed pulse output, communication, MTR, HKY, DSW, SEGL and PR instructions.

Variable address registers D8028,D8029 D8182~D8195

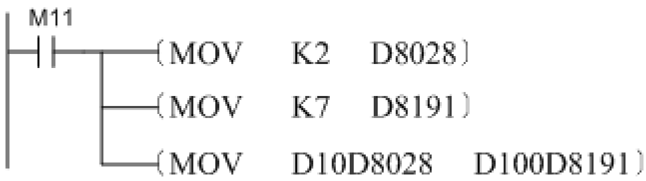
The variable address registers can be used in the same way as ordinary data registers, and they can also be used in the operands of application instructions in cooperation with other software component number and value. But please bear in mind that the software component number of basic sequential instructions such as LD, AND and OUT etc. and step ladder diagram instructions can't be used with the variable address registers. Z or V is usually used as variable address modifier in the program :



Z and V also can also be used as ordinary registers.



Z and V have corresponding special registers, so the first program is equivalent to



D8028 is the content of Z0 register; D8029 is the content of V0 register;
D8182 is the content of Z1 register; D8183 is the content of V1 register;
D8184 is the content of Z2 register; D8185 is the content of V2 register;
D8186 is the content of Z3 register; D8187 is the content of V3 register;
D8188 is the content of Z4 register; D8189 is the content of V4 register;
D8190 is the content of Z5 register; D8191 is the content of V5 register;
D8192 is the content of Z6 register; D8193 is the content of V6 register;
D8194 is the content of Z7 register; D8195 is the content of V7 register;

Bit state M8031~M8034

1. When M8031 is ON all the registers and relays which can't hold their status during power cut will be cleared;
contact state of Y component, generally used M component and generally used S component

contact and timer coil of generally used T component
contact, timer coil and reset coil of generally used C component
current value register of generally used D component
current value register of generally used T component
current value register of generally used C component

2. When M8032 is ON all the registers and relays which can hold their status during power cut will be cleared;
contact state of M and S components which can hold their status during power cut
contact and timer coil of accumulative timer T component
contact and counter coil of C which can hold its status during power cut and high speed counter C component
current value register of D component which can hold its status during power cut
current value register of accumulative timer T component
current value register of C component which can hold its status during power cut and high speed counter C component

3. When M8033 is ON all the software components which is in stop state will remain the same state as before the operation
All the contact states of Y, M and S component
All the contacts and timer coil of T component
All the contacts and timer coil of C component and high speed C component
All the current value register of D, T and C component

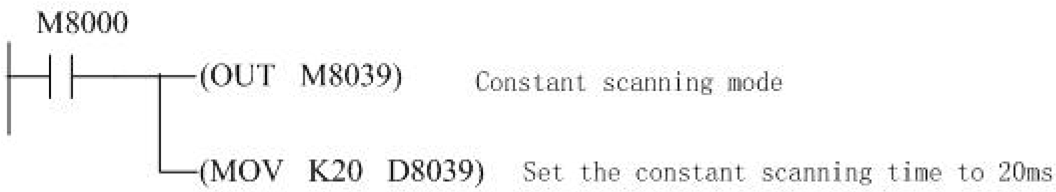
4. When M8034 is ON all the output points of Y component become OFF.

5. Compulsive RUN/STOP operation can make M8035 (compulsive RUN mode) and M8036 (compulsive RUN) change to ON so as to start the PLC.

6. Setting M8037 (compulsive STOP) to ON can stop the operation of PLC.

Constant scan mode M8039 D8039

Drive the auxiliary relay M8039 and then write the target scan time (ms) into data register D8039 in advance, so the operating period of the PLC will not be lower than this value. Even the operation completed ahead of time the system will wait in the remaining time and return to step 0 only when the scan time equals to the value of D8039. If the D8039 value is less than the actual scan time of the program then the system relies on the latter one.



The application of RAMP, HKY, SEGL, ARWS and PR instructions which are relative to scan time must use "constant scan time mode" or cooperates with "on-time interrupt insertion". Especially the HKY instruction which implements input of 16 digital buttons through 4x4 matrix, its scan time must be fixed to more than 20ms. The displayed scan time of D8010~D8012 also contains constant scan time.

Stepper ladder

stepper M8040~M8049 D8040~D8049

1.When M8040 is ON the STL transfer is disabled and the stop state output will continue.
2.The flag M8041 which is used by the IST instruction when step starts.
3.a pulse output is given in response to a start input, the IST instruction uses flag M8042
4.The flag M8043 which is used by IST instruction will be ON during the last state of ZERO RETURN mode.
5.The flag M8044 which is used by IST instruction will be ON when machine zero is detected.
6.The flag M8045 which is used by IST instruction will be ON when "all output reset" function is disabled.
7.When M8047 is ON and any of S0~S999 is ON then M8046 will be closed automatically. This is used to avoid collision with other processes or as a start flag of working procedure.
8.When M8049 is ON and any of S900~S999 is ON then M8048 will be closed automatically. This is used to avoid collision with other processes or as a start flag of working procedure.
9.D8040~D8047: The minimum alram address of S0~S999 is stored in D8040, and other addresses are stored in turn. The maximum alarm address is stored in M8047.
10¡¢When M8049 is ON ,the minimum alarm address of S0~S999 is stored in D8049.

Interrupt disable

interrupt disable M8050~M8059

If the interrupt is disabled then the interrupt will not be generated even if there comes the interrupt signal. For example, when M8050 is ON the I00 port will not output even if there is interrupt pulse input to the X0 port. The corresponding interrupts are defined respectively as following:

Interrupt enable/disable setting			
M8050	Drive I00□ interrupt disabled	X input interrupt, 12 interrupts in total, respectively corresponding to rising edge interrupts and falling edge interrupts. In □: 0= rising edge interrupts 1= falling edge interrupts	Each flag controls one external interrupt. When M flag is set to OFF, related X interrupt is enabled. When M flag is set to ON, related X interrupt is disabled.
M8051	Drive I10□ interrupt disabled		
M8052	Drive I20□ interrupt disabled		
M8053	Drive I30□ interrupt disabled		
M8054	Drive I40□ interrupt disabled		
M8055	Drive I50□ interrupt disabled		
M8056	Drive I6□ interrupt disabled	Timing interrupt 0	
M8057	Drive I7□ interrupt disabled	Timing interrupt 1	
M8058	Drive I8□ interrupt disabled	Timing interrupt 2	
M8059	Drive counter interrupt disabled	High speed counter interrupt, 6 in total	ON to disable interrupt I010-I060.

The pulse capture function of X0~X5 is not restricted by interrupt disable operations.

Link operation function

parallel link protocol M8070~M873 M8162 D8070

M8070 will be driven when the PLC is a master station in a parallel link application. M8071 will be driven when the PLC is a slave station in a parallel link application. The M8070 and M8071 can't be driven simultaneously in one PLC, or the parallel link protocol will be invalid. The parallel link protocol can also be setup through D8126 if there aren't any other high level protocols. Setting D8126 to 50h can set the PLC to a master station in a parallel link application, and setting D8127 to 5h can set the PLC to a slave station.

D8070: time setting of communication error detection
M8162: high speed parallel link mode

	Master station TX (slave station RX)	slave station Tx (Master station Tx)
Normal mode M8162=0	M800~M899 D490~D499	M900~M999 M500~M509
High speed mode M8162=1	D490~D491	D500~D501

Please refer to the introduction to parallel link protocol in the communication part for specific function setting

N:N protocol D8076~D8180 M8183~M8191 D8201~D8218

The users just need to set one PLC to N:N protocol master station and set many other PLCs to N:N protocol slave station, then connect all the PLCs through serial port, the multiple PLCs can exchange data without intervention of user program.

D8176: station number£¬range 0~7,0 means master station
D8177: total number of slave stations, range 1~7,only needed by master station
D8178: refresh range(mode) setting, range 0~2,only needed by master station
D8179: retry count setting, only needed by master station
D8180: communication overtime setup,*10ms,only needed by master station
M8183~M8190: data transfer master station error,M8183 corresponds to station 0(master station),M8184 corresponds to slave station 1, and so forth,M8190 corresponds to slave station 7.
M8191: data transferring
D8201: current connection scan time
D8202: maximum connection scan time
D8203: master station communication error number, counting stop after reaching maximum value 10000
D8204~ D8210: slave station 1~7 communication error number,counting stop after reaching maximum value 10000
D8211: master station communication error code
D8212~ D8218: respective slave station 1~7 communication error code. Error message description can be found out according to D8211~ D8218 error code.

Please refer to the introduction to N:N protocol in the communication part for specific function setting.

Special function

output initialization M8085~M8089

Setting special bits M8085~M8089 (corresponds to Y0~Y4 respectively) to ON can enable following functions in PLSY,PLSR,DRVI and DRVA instructions:

Driving special bits to ON can start next pulse output instruction immediately without the need of invalid processing of last power flow;

output finished interrupt M8090~M8094

Setting special bits M8090~M8094 (corresponds to Y0~Y4 respectively) to ON can enable following functions in PLSY, PLSR,DRVI and DRVA instructions:

The output finished interrupt can be enabled; Following is the details:

Port no.	Special bit	Related user interrupt
Y0	M8090	I502
Y1	M8091	I503
Y2	M8092	I504
Y3	M8093	I505
Y4	M8094	I506

Please refer to the introduction to H2U special functions for specific function setting

acceleration/deceleration time M8135~M8139 D8104~D8108 D8165~D8169

Setting special bits M8135~M8139 (corresponds to Y0~Y4 respectively) to ON can enable following functions in PLSY, PLSR,DRVI and DRVA instructions:

The number of output pulses can be changed during operation (larger or smaller, and the number can only be changed during acceleration and uniform motion, and if it is changed during deceleration then it's invalid)

Deceleration time is defined by following registers in PLSR,DRVI and DRVA instructions:

Port no.	Special bit	Related register
Y0	M8135	D8165
Y1	M8136	D8166
Y2	M8137	D8167
Y3	M8138	D8168
Y4	M8139	D8169

When M8135 is ON the deceleration time of YO is determined by D8165, and the default value is 100ms. And so forth.

Acceleration time is defined by following registers in DRVI and DRVA instructions:

Port no.	Special bit	Related register
Y0	M8135	D8104
Y1	M8136	D8105
Y2	M8137	D8106
Y3	M8138	D8107
Y4	M8139	D8108

When M8135 is ON the acceleration time of YO is determined by D8104, and the default value is

100ms. And so forth.

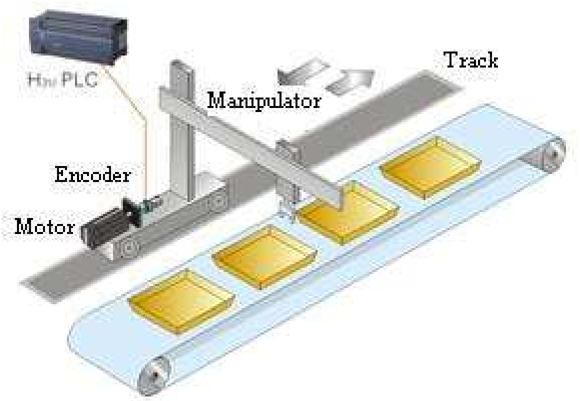
Please refer to the introduction to H2U special functions for specific function setting

high speed multiple user interrupt M8084 D8084~D8086

Any high speed counter (C235~C255) can be set to generate multiple interrupts to support multiple high speed free task during the operation of high speed counter. This function is called high speed multiple user interrupts. Its maximum number is 24;

Flag	Discretion
M8084	ON to enable high speed counter multi-user interrupts
D8084	High speed counter no. C235~255
D8085	Related user interrupt number, up to 24 from I507 to I530
D8086	The series number corresponding to multiple point data should be D double-word component, such as 200 is the double-word started with D200.

For example: When the traveling crane is carrying out load/unload task, it's needed to carry out different task on different locations. So using the high speed multiple user interrupt processing function has the advantages of fast response and easy operation.



Please refer to the introduction to H2U special functions for specific function setting

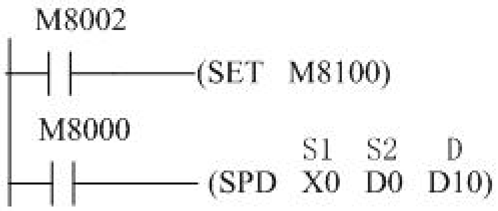
Meter counter and tachometer M8100~M8105

If the function enable flag M8100~M8105 (corresponds to X0~X5 respectively) is ON in the SPD instruction, then the definition of D component in the SPD S1 S2 D will be different from original one,

When the corresponding M8100~M8105 is OFF:
D+0: is the pulse count during S2 time and it¡¯s a 16 bit data; D+1: is the pulse count during this time segment; D+2: is used for measuring remaining time(MS).

When the corresponding M8100~M8105 is ON:
D+0: is the pulse count during S2 time and it's a 16 bit data; D+1,D+2: is the pulse count per minute and it's a 32 bit data;

example instruction



X0 is input port;
D0 is time unit (ms);
D10 is total number of pulse during D0 time segment;
D11、D12 are operating frequency= the number of pulse in one minute *10 (unit: 0.1)

Please refer to the introduction to H2U special functions for specific function setting

output port reset M8140 D8156~D8160

When M8140 is ON the output point corresponding to D8156~D8160 will be ON for one scan period after the system returned to zero by ZRN instruction, which can be used for CLR servo control.

D8156:YO port reset signal definition, default 5= Y05
D8157:Y1 port reset signal definition, default 6= Y06
D8158:Y2 port reset signal definition, default 7= Y07
D8159:Y3 port reset signal definition, default 8= Y10
D8160:Y4 port reset signal definition, default 9= Y11

High speed ring counter

high speed ring counter M8099 D8099

After M8099 is driven the special data register D8099 will increase 0.1ms accumulatively from the next scan period. The value of D8099 will restart from 0 when it exceeds 32767. The pulse width which takes 1ms or 0.1ms as unit can be measured through accumulative type 1ms timer or special data register D8099 (high speed ring counter).

Communication and link

COM0:D8110~D8119

COM0 protocol	Set by D8116	Half/full duplex	Communication format
Download protocol/HMI monitor protocol	01h	Set by jumper JPO	7E1 fixed
MODBUS-RTU slave station	02h	Half duplex	Set by D8110
MODBUS-ASC slave station	03h	Half duplex	Set by D8110
Others (including RS instruction)	Not supported		

COM1:M8121~M8124 M8129 D8110~D8119

COM1 protocol	Set by D8116	Half/full duplex	Communication format
RS instruction	00h	Set by bit 10 in D8120*	Set by D8120
HMI monitor protocol	01h	Half duplex	Fixed
Parallel protocol master station	50h	Half duplex	Fixed
Parallel protocol slave station	05h	Half duplex	Fixed
N:N protocol master station	40h	Half duplex	Fixed
N:N protocol slave station	04h	Half duplex	Fixed
Computer link protocol	06h	Half duplex	Set by D8120
MODBUS-RTU slave station	02h	Half duplex	
MODBUS-ASC slave station	03h	Half duplex	
RS instruction	10h	Set by bit 10 in D8120*	
MODBUS-RTU slave station	20h	Half duplex	
MODBUS-ASC slave station	30h	Half duplex	

*half duplex/ full duplex mode of RS instruction can be set by Bit10 of D8120:

1:half duplex RS485 (standard port)
0:full duplex RS232C/RS422 (extension board H2U-232BD or H2U-422BD is needed)

M8120	Reserved	D8120	Communication format, the interface configuration with a default of 0
M8121	Sending and waiting (RS instruction)	D8121	Station number settings, the interface configuration settings with a default of 1
M8122	Sending flag (RS instruction) Instruction execution status (MODBUS)	D8122	Amount of remaining data to be transmitted (Only to RS instruction)
M8123	Receiving complete flag (RS) Communication error flag (MODBUS)	D8123	Amount of data already received (Only to RS instruction)
M8124	Receiving (only to RS instruction)	D8124	Start character STX (Only to RS instruction)
M8125	Reserved	D8125	Termination character ETX (Only to RS instruction)
M8126	If M8126 is ON, 485BD extended card available	D8126	Communication protocol, the interface configuration with a default of 0
M8127	Reserved	D8127	Computer link protocol of data starting address
M8128	Reserved	D8128	Computer link protocol sending data amount
M8129	Timeout judgement	D8129	Communication overtime judgement, the interface configuration settings with a default of 10 (100ms)

function difference between COM0 and COM1

COM0 hardware is standard RS485 and RS422 which are compatible with each other, and selection can be made through jumper. The connection terminal is a female 8-pins PS/2 connector.
COM1 hardware is RS485 and the terminal is terminal block.
The 485 of COM0 can only be slave station which doesn't support RS instruction, linking function and computer link protocol and so on;
The 485 of COM1 can be master or slave station, and it supports RS instruction, linking function and computer link protocol and so on;

Positioning

Positioning M8145~M8154 D8136~D8160

M8145: When the M8145 is set to ON during the pulse output process, Y0 will stop the output pulse immediately, and the PLSY and DRVI instruction will output pulses from zero when conducting the next power flow. The DRVA instruction can output the remaining pulses when conducting the next power flow.
M8146: When the M8146 is set to ON during the pulse output process, Y1 will stop the output pulse immediately, and the PLSY and DRVI instruction will output pulses from zero when conducting the next power flow. The DRVA instruction can output the remaining pulses when conducting the next power flow.
M8152: When the M8152 is set to ON during the pulse output process, Y2 will stop the output pulse immediately, and the PLSY and DRVI instruction will output pulses from zero when conducting the next power flow. The DRVA instruction can output the remaining pulses when conducting the next power flow.
M8153: When the M8153 is set to ON during the pulse output process, Y3 will stop the output pulse immediately, and the PLSY and DRVI instruction will output pulses from zero when conducting the next power flow. The DRVA instruction can output the remaining pulses when conducting the next power flow.

M8154: When the M8154 is set to ON during the pulse output process, Y4 will stop the output pulse immediately, and the PLSY and DRVI instruction will output pulses from zero when conducting the next power flow. The DRVA instruction can output the remaining pulses when conducting the next power flow.
M8147: If YO has pulse output, then M8147 will be ON, and if the pulse output stops then M8147 will be OFF. So this can be used for monitoring;
M8148: If Y1 has pulse output, then M8148 will be ON, and if the pulse output stops then M8148 will be OFF. So this can be used for monitoring;
M8149: If Y2 has pulse output, then M8149 will be ON, and if the pulse output stops then M8149 will be OFF. So this can be used for monitoring;
M8150: If Y3 has pulse output, then M8150 will be ON, and if the pulse output stops then M8150 will be OFF. So this can be used for monitoring;
M8151: If Y4 has pulse output, then M8151 will be ON, and if the pulse output stops then M8151 will be OFF. So this can be used for monitoring;

D8136: Low word; D8137: High word
Used as current value data accumulator register of the Y0 and Y1 output positioning instructions; they correspond to current increasing /decreasing value of the rotation direction. The PLSY and PLSR instructions, which only have a pulse output but no direction signal, use the same current value data registers; therefore, the current value is the cumulative value of the pulse numbers when these instructions are executed;

D8140: Low word ; D8141: High word
Used as current value data register of the Y0 output positioning instructions; they correspond to current increasing /decreasing value of the rotation direction. The PLSY and PLSR instructions, which only have a pulse output but no direction signal, use the same current value data registers; therefore, the current value is the cumulative value of the pulse numbers when these instructions are executed;

D8142: Low word ; D8143: High word
Used as current value data register of the Y1 output positioning instructions; they correspond to current increasing /decreasing value of the rotation direction. The PLSY and PLSR instructions, which only have a pulse output but no direction signal, use the same current value data registers; therefore, the current value is the cumulative value of the pulse numbers when these instructions are executed;

D8150: Low word ; D8151: High word
Used as current value data register of the Y2 output positioning instructions; they correspond to current increasing /decreasing value of the rotation direction. The PLSY and PLSR instructions, which only have a pulse output but no direction signal, use the same current value data registers; therefore, the current value is the cumulative value of the pulse numbers when these instructions are executed;

D8152: Low word ; D8153: High word
Used as current value data register of the Y3 output positioning instructions; they correspond to current increasing /decreasing value of the rotation direction. The PLSY and PLSR instructions, which only have a pulse output but no direction signal, use the same current value data registers; therefore, the current value is the cumulative value of the pulse numbers when these instructions are executed;

D8154: Low word; D8155: High word
Used as current value data register of the Y4 output positioning instructions; they correspond to current increasing /decreasing value of the rotation direction. The PLSY and PLSR instructions, which only have a pulse output but no direction signal, use the same current value data registers;

therefore, the current value is the cumulative value of the pulse numbers when these instructions are executed;

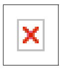
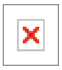
D8145: the basic speed when executing DRVI and DRVA instructions. Speed setting should take the resonant area and self-start frequency of the stepper motor into consideration when controlling stepper motors. Setting range: lower than 1/10 of the maximum speed (D8l47, D8l46). If the setting value were to exceed this range, then the actual speed will be 1/10 of the maximum speed.

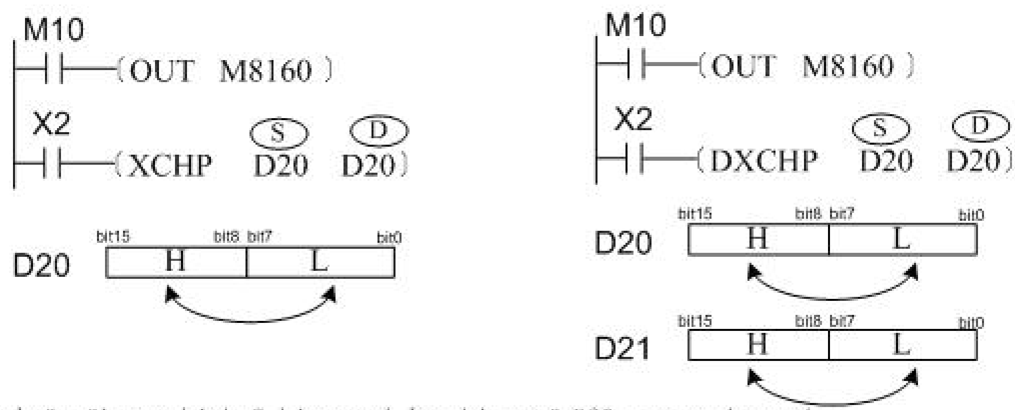
D8146: Low word ; D8147: High word
This is the maximum speed when executing DRVI and DRVA instructions. The pulse frequency of the high speed output port should not exceed this speed. Setting range: 10~100, 000£¨Hz£©

D8l48: the acceleration/deceleration time when executing DRVI, DRVA, and PLSR instructions, meaning the time needed to reach the maximum speed (D8147, D8146). If the output pulse frequency is lower than the maximum speed (D8147 , D8146), then the actual acceleration/deceleration time will be shortened. Setting range: 50 ~ 5, 000 (ms) ;when M8135~M8140

Data processing

SWAP function of XCH M8160

When M8160=1 and  have the same address as , then the high byte and low byte will be exchanged. For 32-bit instruction, the operation will exchange the high byte and low byte of the two (2) registers respectively. This corresponds to the SWAP instruction and generally can be implemented using SWAP instruction.



In left figure high 8 bits and low bits of D20 are exchanged.

In right figure high 8 bits and low bits of D20 are exchanged.
High 8 bits and low bits of D21 are exchanged.

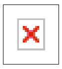
8 bit and 16 bit mode M8161

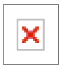
M8161 flag determines the width mode of variables. If M8161 = OFF then the variables are 16 bit mode; If M8161 = ON then the variables are 8 bit mode, so the actual length of variable area will increase. This can be used in ASC/RS/ASCII/HEX/CCD.

move points variable mode M8164 D8164

FROM/TO move points variable mode: If M8164 = ON then the value of special data register D8164 (move points register of FROM/TO instruction) will be treated as move points n;

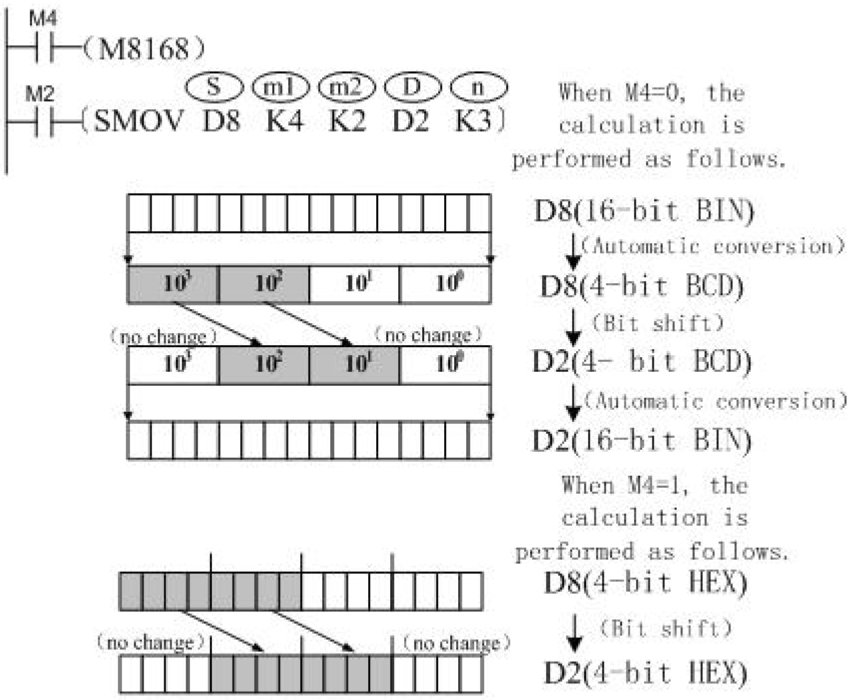
bit to word function M8167

If M8167 is set to ON in HEY instruction then HEY instruction will store 0~F keys as hexadecimal data to  unit memory.

For example: In  [123BF] is stored as BIN format [123BF] , so as to change the function of A~F, please refer to HEY instruction for details.

SMOV decimal and hex data switch function M8168

In the SMOV instruction, if M8168 is OFF then the data is BCD mode (decimal) and if M8168 is ON then the data is BIN mode in which 4 bits are transferred as one unit (hexadecimal).



Pulse capture

pulse capture M8170~M8175

M8170 X00 pulse capture
M8171 X01 pulse capture
M8172 X02 pulse capture
M8173 X03 pulse capture
M8174 X04 pulse capture
M8175 X05 pulse capture
The "pulse capture" function can be used when response to instantaneous pulse signal at X0~X5 ports is needed without special requirement of response time. PLC will store the rising edge signal of X0~X5 ports to M8170~M8175 which can be used by main routine to judge and process and can be cleared manually after response. After executing interrupt enable EI instruction the auxiliary relay

M8170 ~ M8175 will be set to process interrupts when the input ports X000~X005 change from OFF→ON. Reset to preset components must be carried out by program to acquire pulse again. The pulse capture action is independent of auxiliary relays M8050~ M8055 which are used by interrupt disable instruction, which means that setting M8050~ M8055 to ON can't disable the pulse capture function.

High speed input

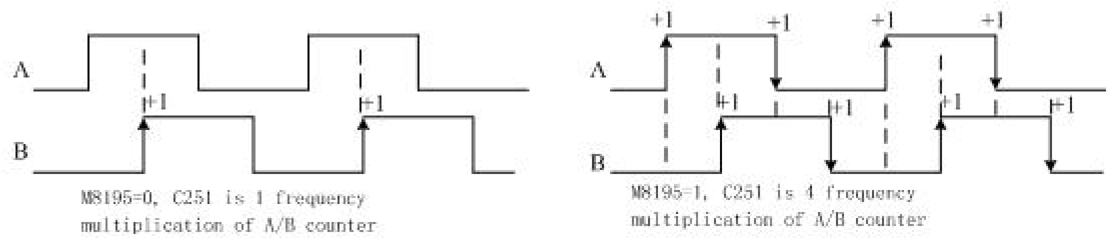
high speed table comparison mode M8130 M8131 D8130

In the high speed table comparison mode, when the data of first row is the same then the table counter D8130 changes to 1 and proceed to the second row and so forth. After the last row the operation finished flag M8131 actuated and returns to the initial row. Please refer to for specific application.

double-frequency control M8195~M8199

A/B phase high speed counter T251~T255 has two frequency modes of 1 double-frequency and 4 fold frequency which is defined respectively by special registers M8195¡«M8199. The signal of AB phase high speed counter occupies two pulse input ports and the equivalent pulse number of PLC will be multiplied by 2. If the A/B input mode of C251~C255 is double-frequency mode then the maximum high speed input frequency is 50kHz. If the A/B input mode of C251~C255 is 4 fold frequency mode then the counter is in software mode and the maximum high speed input frequency is 25kHz.

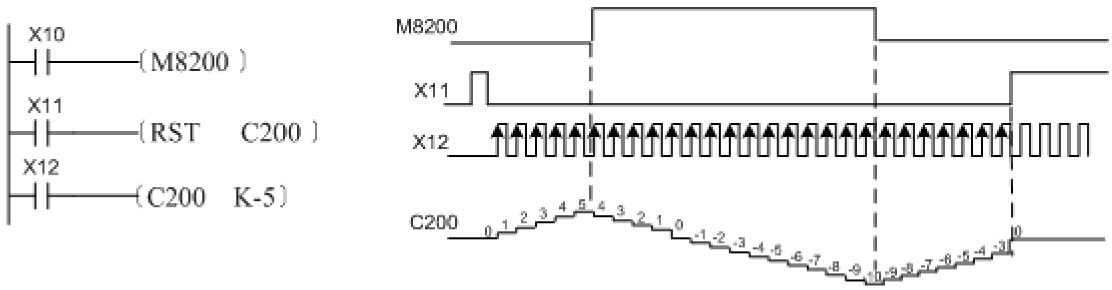
If M8195 is OFF then the AB phase input of C251 is one double frequency; If M8195 is ON then the AB phase input of C251 is four fold frequency mode, which is shown in following figure:



It's the same as C251:
If M8196 is OFF then the AB phase input of C252 is one double frequency mode; If M8196 is ON then the AB phase input of C252 is four fold frequency mode.
If M8197 is OFF then the AB phase input of C253 is one double frequency mode; If M8197 is ON then the AB phase input of C253 is four fold frequency mode.
If M8198 is OFF then the AB phase input of C254 is one double frequency mode; If M8198 is ON then the AB phase input of C254 is four fold frequency mode.
If M8199 is OFF then the AB phase input of C255 is one double frequency mode; If M8199 is ON then the AB phase input of C255 is four fold frequency mode.

32 bit up/down counter control M8200~M8234

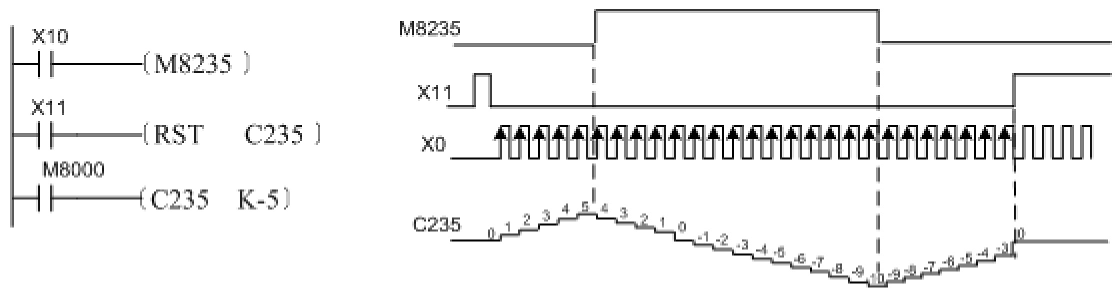
Special auxiliary relays M8200~M8234 can be used to define up/down counter direction for general 32 bit up/down counters. If C*** drives M8*** then the counter will count down, or the counter will count up, for example:



If M8200 is OFF then the C200 will count up; If M8200 is ON then the C200 will count down. It's the same as C200:
If M8201 is OFF then the C201 will count up; If M8201 is ON then the C201 will count down.
If M8202 is OFF then the C202 will count up; If M8202 is ON then the C202 will count down.
If M8203 is OFF then the C203 will count up; If M8203 is ON then the C203 will count down.
If M8204 is OFF then the CC204 will count up; If M8204 is ON then the C204 will count down.
If M8205 is OFF then the C205 will count up; If M8205 is ON then the C205 will count down.
If M8206 is OFF then the C206 will count up; If M8206 is ON then the C206 will count down.
If M8207 is OFF then the C207 will count up; If M8207 is ON then the C207 will count down.
If M8208 is OFF then the C208will count up; If M8208 is ON then the C208 will count down.
If M8209 is OFF then the C209 will count up; If M8209 is ON then the C209 will count down.
If M8210 is OFF then the C210 will count up; If M8210 is ON then the C210 will count down.
If M8211 is OFF then the C211 will count up; If M8211 is ON then the C211 will count down.
If M8212 is OFF then the C212 will count up; If M8212 is ON then the C212 will count down.
If M8213 is OFF then the C213 will count up; If M8213 is ON then the C213 will count down.
If M8214 is OFF then the C214 will count up; If M8214 is ON then the C214 will count down.
If M8215 is OFF then the C215 will count up; If M8215 is ON then the C215 will count down.
If M8216 is OFF then the C216 will count up; If M8216 is ON then the C216 will count down.
If M8217 is OFF then the C217 will count up; If M8217 is ON then the C217 will count down.
If M8218 is OFF then the C218 will count up; If M8218 is ON then the C218 will count down
If M8219 is OFF then the C219 will count up; If M8219 is ON then the C219 will count down.
If M8220 is OFF then the C220 will count up; If M8220 is ON then the C220 will count down.
If M8221 is OFF then the C221 will count up; If M8221 is ON then the C221 will count down.
If M8222 is OFF then the C222 will count up; If M8222 is ON then the C222 will count down.
If M8223 is OFF then the C223 will count up; If M8223 is ON then the C223 will count down.
If M8224 is OFF then the C224 will count up; If M8224 is ON then the C224 will count down.
If M8225 is OFF then the C225 will count up; If M8225 is ON then the C225 will count down.
If M8226 is OFF then the C226 will count up; If M8226 is ON then the C226 will count down.
If M8227 is OFF then the C227 will count up; If M8227 is ON then the C227 will count down.
If M8228 is OFF then the C228 will count up; If M8228 is ON then the C228 will count down.
If M8229 is OFF then the C229 will count up; If M8229 is ON then the C229 will count down.
If M8230 is OFF then the C230 will count up; If M8230 is ON then the C230 will count down.
If M8231 is OFF then the C231 will count up; If M8231 is ON then the C231 will count down.
If M8232 is OFF then the C232 will count up; If M8232 is ON then the C232 will count down.
If M8233 is OFF then the C233 will count up; If M8233 is ON then the C233 will count down.
If M8234 is OFF then the C234 will count up; If M8234 is ON then the C234 will count down.

high speed single phase up/down counter control M8235~M8245

The single phase high speed counter input has only 1 counter pulse input port, and the program determines the count direction through respective special M registers.

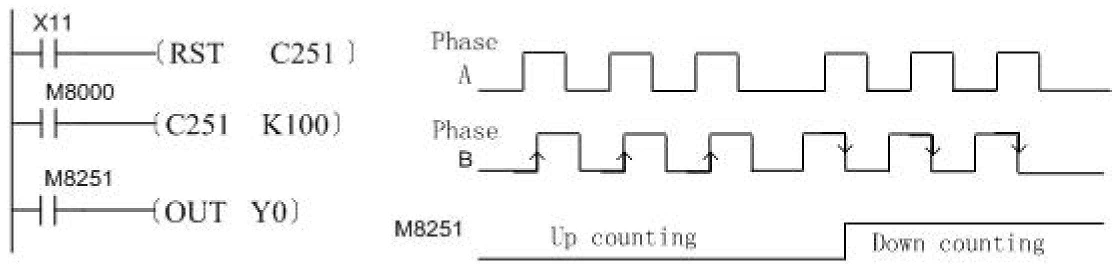


If M8235 is OFF then the C235 will count up; If M8235 is ON then the C235 will count down. It's the same as C235:
If M8236 is OFF then the C236 will count up; If M8236 is ON then the C236 will count down.
If M8237 is OFF then the C237 will count up; If M8237 is ON then the C237 will count down.
If M8238 is OFF then the C238 will count up; If M8238 is ON then the C238 will count down.
If M8239 is OFF then the C239 will count up; If M8239 is ON then the C239 will count down.
If M8240 is OFF then the C240 will count up; If M8240 is ON then the C240 will count down.
If M8241 is OFF then the C241 will count up; If M8241 is ON then the C241 will count down.
If M8242 is OFF then the C242 will count up; If M8242 is ON then the C242 will count down.
If M8243 is OFF then the C243 will count up; If M8243 is ON then the C243 will count down.
If M8244 is OFF then the C244 will count up; If M8244 is ON then the C244 will count down.
If M8245 is OFF then the C245 will count up; If M8245 is ON then the C245 will count down.
C241~C245 have hardware reset input function and some of them have hardware start/stop input control function.

high speed up/down counter control M8246~M8255

Single phase 2 input high speed counter and AB phase input high speed counter can automatically count up or down through external input.

The single phase 2 input counter has two pulse input ports which are respectively count-up pulse input port and count-down pulse input port; Some counters have signal input port for hardware reset and start/stop; AB phase counter determines counter direction by AB phases: If A pulse is high level then the rising edge of B pulse causes counting up and the falling edge causes counting down.
The up/down counting status of C246-C255 can be monitored by reading out the states of M8246-M8255. Following figure shows the monitoring of C251:



If C251 counts up then M8251 is OFF; If C251 counts down then M8251 is ON. Monitoring M8251 can indicate that whether C251 counts up or down. It¡¯s the same as C251:
If C246 counts up then M8246 is OFF; If C246 counts down then M8246 is ON. Monitoring M8246 can indicate that whether C246 counts up or down.
If C247 counts up then M8247 is OFF; If C247 counts down then M8247 is ON. Monitoring M8247 can indicate that whether C247 counts up or down.
If C248 counts up then M8248 is OFF; If C248 counts down then M8248 is ON. Monitoring M8248 can indicate that whether C248 counts up or down.
If C249 counts up then M8249 is OFF; If C249 counts down then M8249 is ON. Monitoring M8249

can indicate that whether C249 counts up or down.
If C250 counts up then M8251 is OFF; If C250 counts down then M8250 is ON. Monitoring M8250 can indicate that whether C250 counts up or down.
If C252 counts up then M8252 is OFF; If C252 counts down then M8252 is ON. Monitoring M8252 can indicate that whether C252 counts up or down.
If C253 counts up then M8253 is OFF; If C253 counts down then M8253 is ON. Monitoring M8253 can indicate that whether C253 counts up or down.
If C254 counts up then M8254 is OFF; If C254 counts down then M8254 is ON. Monitoring M8254 can indicate that whether C254 counts up or down.
If C255 counts up then M8255 is OFF; If C255 counts down then M8255 is ON. Monitoring M8255 can indicate that whether C255 counts up or down.