EzMove Utility

(Version 3.0)

ICP DAS CO., LTD.

Warranty

All products manufactured by ICPDAS Inc. are warranted against defective materials for a period of one year from the date of delivery to the original purchaser.

Warning

ICPDAS Inc. assumes no liability for damages consequent to the use of this product. ICPDAS Inc. reserves the right to change this manual at any time without notice. The information furnished by ICPDAS Inc. is believed to be accurate and reliable. However, no responsibility is assumed by ICPDAS Inc. for its use, or for any infringements of patents or other rights of third parties resulting from its use.

Copyright

Trademark

The names used for identification only maybe registered trademarks of their respective companies.

License

The user can use, modify and backup this software on a single machine. The user may not reproduce, transfer or distribute this software, or any copy, in whole or in part.

Technical Support

If you have problems about using the product, please contact ICP DAS Product Support.

Email: Service@icpdas.com

1 Introduction

EzMove is a Windows-based utility assisting the user in writing macro programs and in getting familiar with the motion commands provided for the ET-M8194H and RS-M8194H. Furthermore it can be used for motion monitoring and tracking of the motion path. For each command the corresponding Modbus data packet is displayed.

All the configuration and settings options provided by the EzMove utility can also be done by directly calling the corresponding Modbus command or by using the APIs provided by the ET_M8194H_API.dll and RS_M8194H_API.dll.

ET-M8194H is an Ethernet based 4-axis stepping/pulse-type motion controller and uses Modbus TCP/IP as a communication protocol between client and server. The ET-M8194H is designed to be a slave in a MODBUS/TCP network and can be easily integrated into an existing MODBUS/TCP network (for example connected to a PC, PAC, PLC).

RS-M8194H is a RS-232/485/422 serial based 4-axis stepping/pulse-type motion controller and uses Modbus RTU as a communication protocol between master and slave.

1.1 Utility Features

- Remotely controlling the motion controller
- Testing and debugging
- Calling commands directly
- Sending several commands at once (batch)
- Creating, saving and changing Batch files
- Writing macro programs
- Displaying the request and response Modbus data format
- Visualizing the motion
- Getting acquainted with the different commands and their properties
- Motion chip initialization setting
- Displaying FRnet DIO state, daughter board DI state and the motion chip state
- FRnet DI event trigger setting
- Displaying the currently occupied MP and ISR Macro program table

2 Utility Layout

Main user interface

The tree view on the left lists all the motion and macro commands supported by ET-M8194H and RS-M8194H. The middle section of the main window displays the parameters of the selected commands. Here the command parameters have to be set. The right section of the main window forms part of the Macro editing interface. The user has to edit or modify motion commands and their parameter for the different macro programs. The area at the bottom shows the Modbus communication data format.

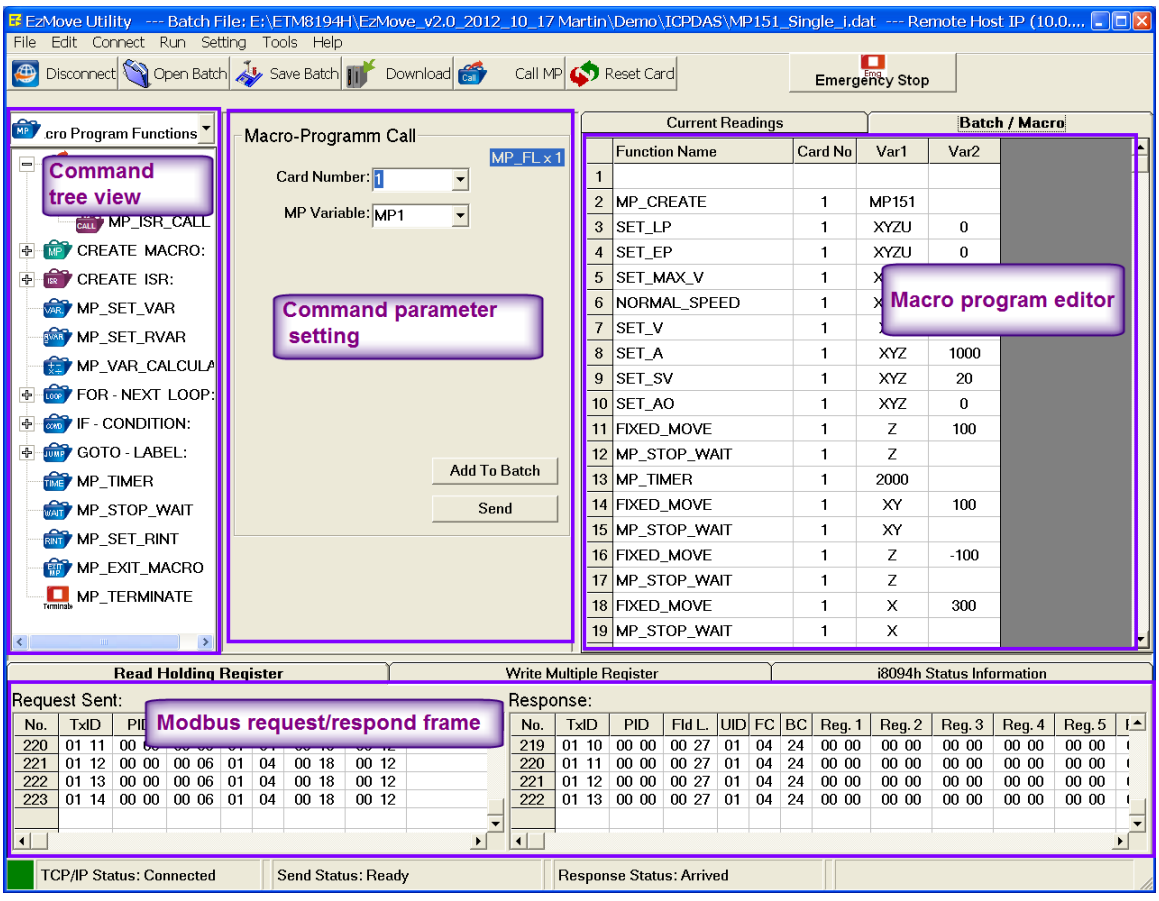
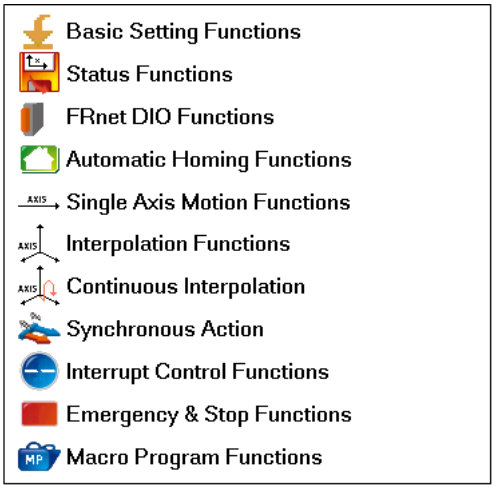


Figure 1: EzMove utility main interface

The main user interfaces of the EzMove utility:

1) Command list

The tree view lists all the commands supported by the ET-M8194H and RS-M8194H. The commands are divided into different categories in order to find them more easily.



2) Parameter input interface
This interface displays the parameters of a selected command and allows the user to assign each parameter a value or a variable. After the commands have been edited the command has to be added to the Macro editor or directly sent to the remote motion controller.

3) Macro/ Batch program editor
The editor can be used for writing
   - Batch programs. A Batch program is made up of a sequence of RTC commands and can be downloaded to a volatile FIFO buffer of the ET-M8194H. When the ET-M8194H/RS-M8194H is in a normal operation mode the RTC commands will immediately start executing one by one once it has been written to the FIFO buffer.
   - Macros programs (MP and ISR) for simple motion and interrupt service routines. Macros are stored in a nonvolatile memory.
   - Single motion command instruction.

   Command selection and parameter setting can be directly done in the editor by clicking the corresponding command field or parameter field. The editor window is resizable by dragging the left and bottom window edge.

4) Modbus data package display
The interface displays for each dispatched motion command the actual Modbus data package sent. The ET-M8194H only accepts Modbus TCP commands and therefore the utility automatically converts the motion command into a Modbus data format before sending it to the ET-M8194H device. For the RS-M8194H the utility converts the motion command into a Modbus RTU commands.

5) Current logical and encoder position display

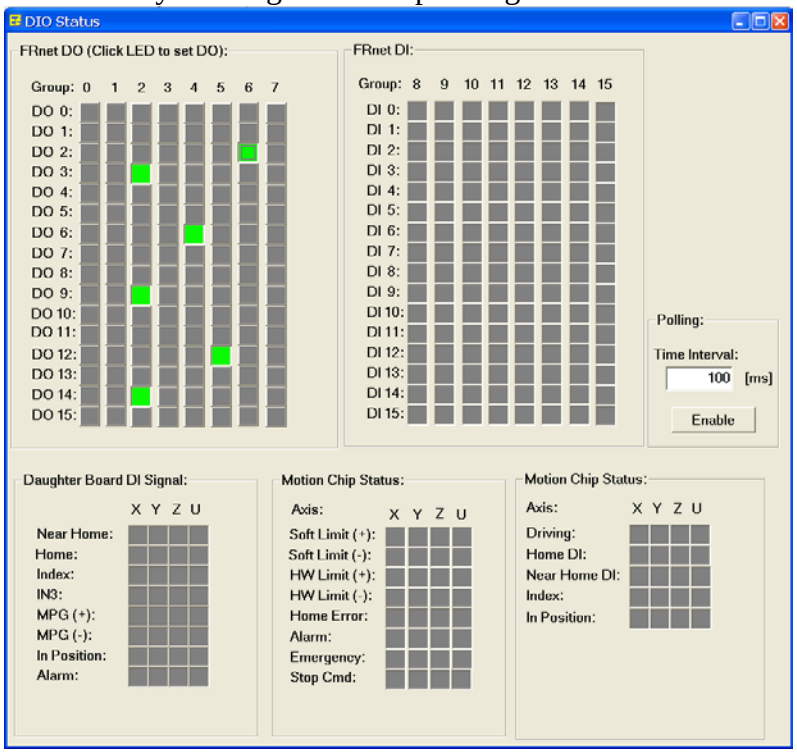
6) Graphical display of the motion pathway in a Cartesian coordinate system

7) Initial table setting window
All the parameter setting for initializing the motion chip after power on can be configured by means of this interface. Furthermore the current initial setting stored in the ET-M8194H will automatically be read by the EzMove utility when the Initial Table Setting window is opened.

8) FRnet DI trigger table
ICPDAS FRnet digital input devices can be used to trigger some events. Each DI channel can be set to trigger a specific event. The supported event settings are as follows: Start, end or hold Macro program execution. This allows the machine operator to do motion control via DI inputs. The FRnet DI interface enables the user to assign each DI channel the trigger condition and the corresponding event.

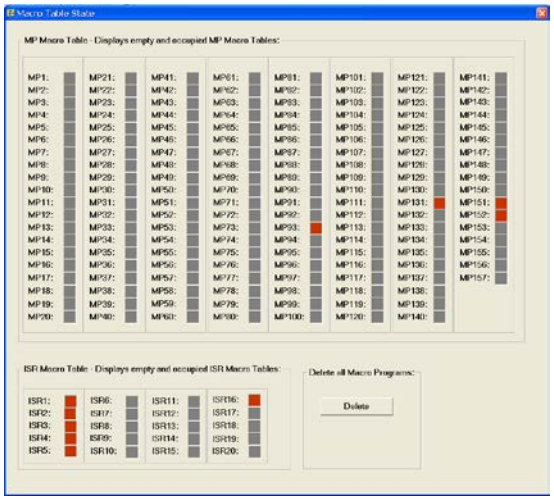
FRnet DI		Setting		
Group	Channel	Trigger Condition	Event Type	Axis
	0	OFF	Pause	
	1	ON to OFF	MP4	
	2	OFF to ON	MP6	
	3	ON to OFF	Hold single axis	YU

9) IO Status
This interface displays the IO status of all the FRnet modules (128 DI and 128 DO), the daughter board and the motion chip state. The individual FRnet DO state can be set by clicking the corresponding DO button.



10) Macro Table State

ET-M8194H support storing up to 157 MP and 20 ISR Macro programs in a non volatile memory. The Macro table state interface indicates which MP and ISR memory already contains a Macro program. This allows the user to keep track of the downloaded macro programs and to accidentally overwrite an existing program.



3 Network Setting

3.1 ET-M8194H

The EzMove utility allows the ET-M8194H network configuration via either RS232 connection or Ethernet.

3.1.1 Setting via RS232

The following steps describe the procedure for setting and reading the network configuration from the ET-M8194H device via COM port.

Step 1: Connect the ET-M8194H module to the COM port of your PC by following Figure 2. (If your PC does not have a COM port ICPDAS provides a USB to RS232 converter: **I-7560**).

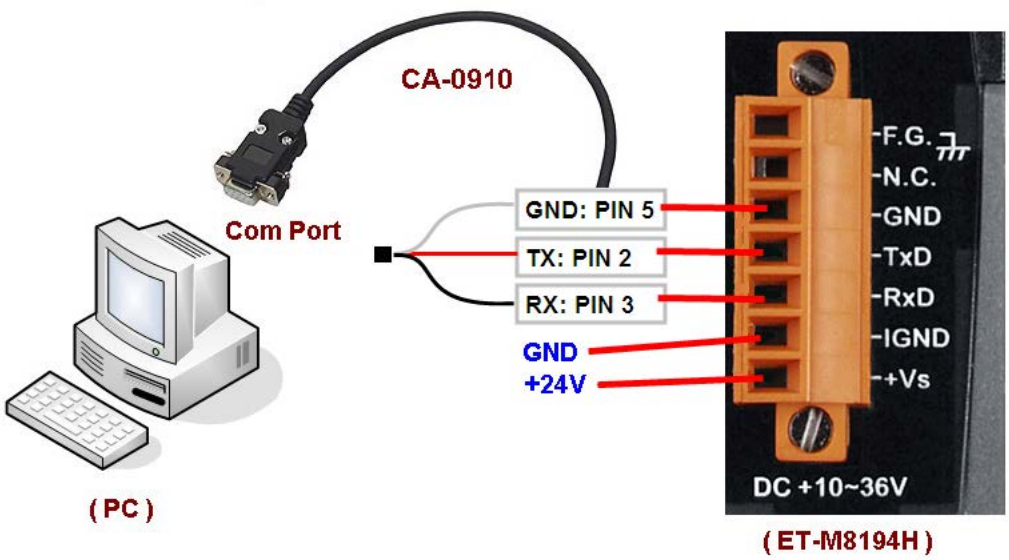


Figure 2: Connecting to COM port

Step 2: Switch the ET-M8149H device **off**.

Step 3: Set the dip switch to "**Init**" (Figure 3).

Figure 3: Set the dip switch to "Init".

Step 4: Switch the ET-M8149H device **on**.

Step 5: Start the EzMove utility.

Step 6: Open the "***Network Setting By COM Port***" dialog window:
Click ***Setting→ET-M8194HSetting→By COM Port→ Network…***

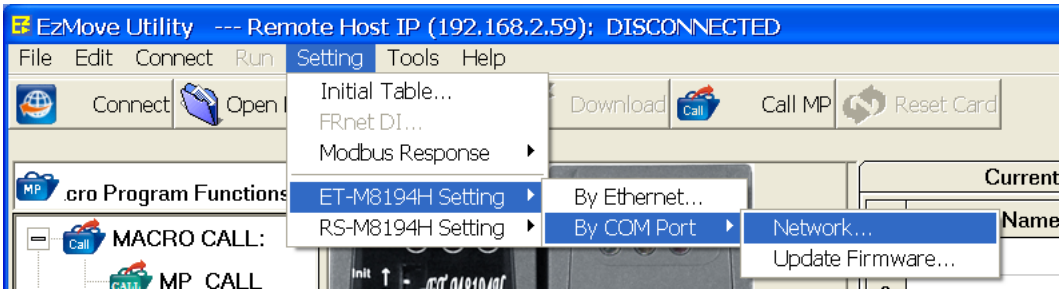
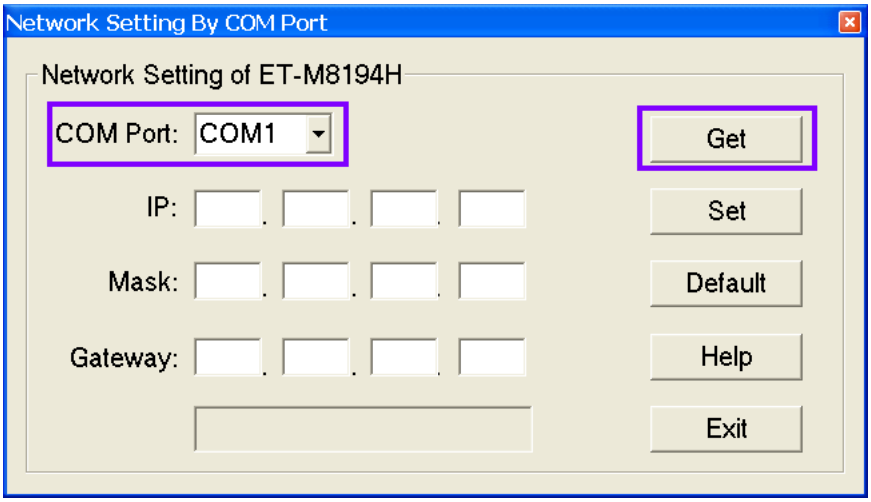


Figure 4: Open network configuration page

Step 7: Select the PC COM port to which the ET-M7184H is connect.



Step 8: Click the "***Get***" button to read the current network setting.

Step 9: Enter IP address, mask and Gateway. Make sure the IP address, subnet mask and gateway are set correctly otherwise the ET-M8194H can not be accessed by the Modbus master or EzMove utility. Click the "***Set***" button to download the new setting to the device.

Step 10: Switch the ET-M8149H device **off**.

Step 11: Set the dip switch to "**Run**" (Figure 5).

(Dip Switch -- Run)

Figure 5: Set the dip switch to "Run".

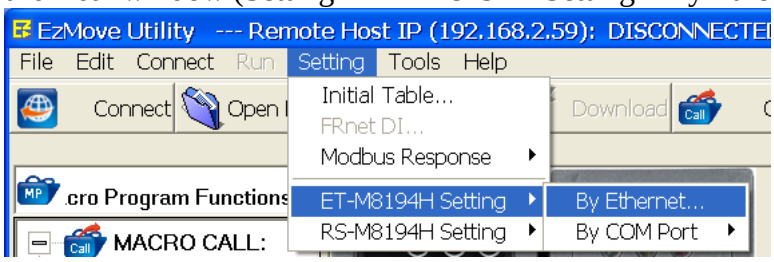
Step 12: Switch on the device. The device can now be accessed via the new network IP address.

Step 13: IMPORTANT!!! The RS232 port of the ET-M8194H is not isolated therefore it is necessary to remove the RS232 cable from the ET-M8194H after the IP setting to prevent electrical noises from occurring.

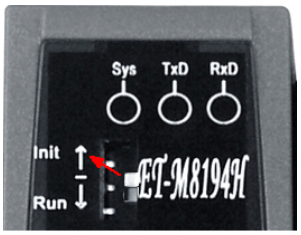
3.1.2 Setting via Ethernet

The following steps describe the procedure for setting and reading the network configuration of the ET-M8194H device via Ethernet:

Step 1: Execute the EzMove Utility and open the "***ET-M8194H Setting by Ethernet***" window (Setting →ET-M8194H Setting→By Ethernet ...).

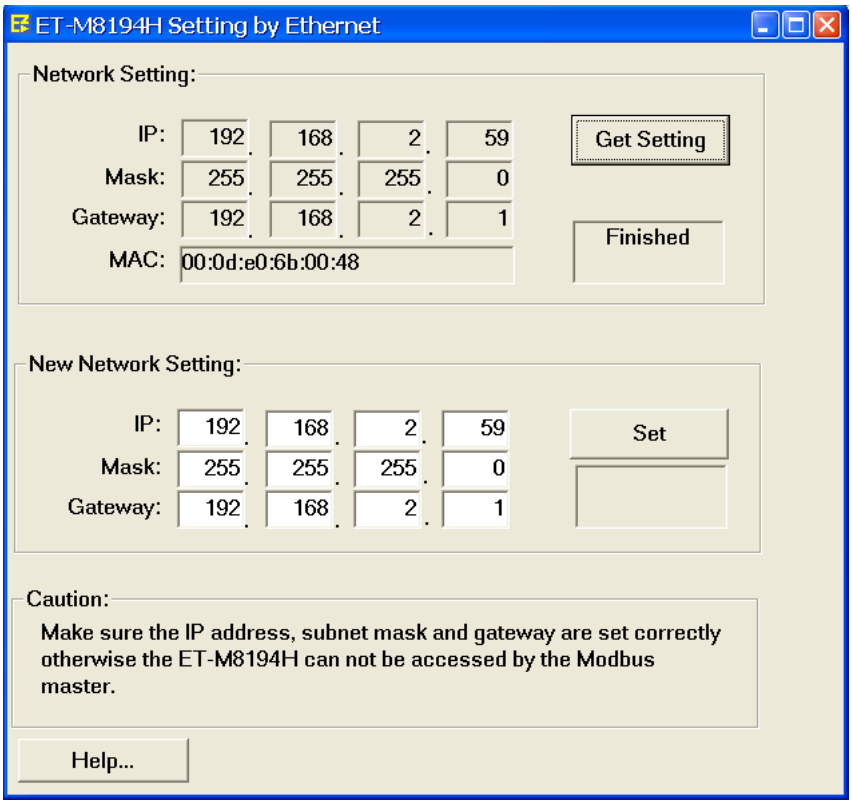


Step 2: Set the DIP-switch to "Init", then power on the ET-M8194H.



(Dip Switch -- Init)

Step 3: Click "***Get Setting***". The utility now reads the Ethernet configuration of the remote ET-M8194H module set into "Init" mode. Make sure that there is only one ET-M8194H module in "Init" mode connected to the Ethernet.



Step 4: Enter a new Ethernet setting and click the "***Set***" button.

Step 5: After the setting is done switch the dip switch back to "Run" and power off/on the module.

(Dip Switch -- Run)

Now the module is ready to be accessed by using the new Ethernet setting.

The ET-M8194H default factory network setting is as follows:

IP:	**192.168.0.16**
Mask:	**255.255.255.0**
Gateway:	**192.168.0.254**

3.2 RS-M8194H

Use the EzMove utility to set the RS-M8194H network configuration.

The following steps describe the procedure for setting and reading the network configuration from the RS-M8194H device via RS232 port.

Step 1: Connect the RS-M8194H module to the RS232 port of your PC by following Figure 6. (If your PC does not have a COM port ICPDAS provides a USB to RS232 converter: **I-7560**).

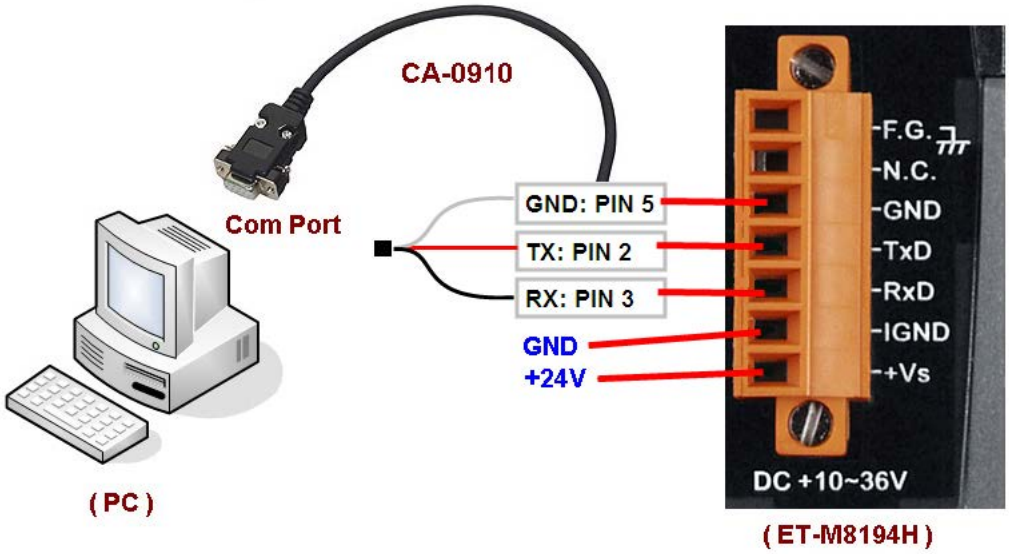


Figure 6: Connecting to COM port

Step 2: Switch the RS-M8149H device **off**.

Step 3: Set the dip switch to "**Init**" (Figure 7).

Figure 7: Set the dip switch to "Init".

Step 4: Switch the RS-M8149H device **on**.

Step 5: Start the EzMove utility.

Step 6: Open the "***RS-M8194H Communication Configuration***" dialog window: Click ***Setting→RS-M8194HSetting→By COM Port→ Commuincation Configuration…***(Figure 8)

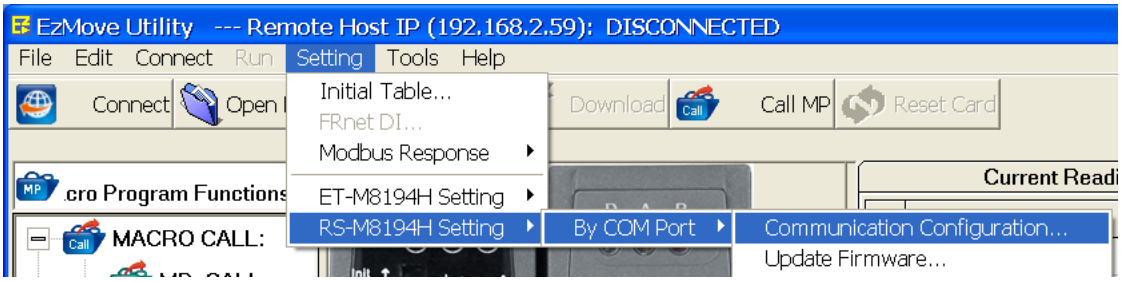
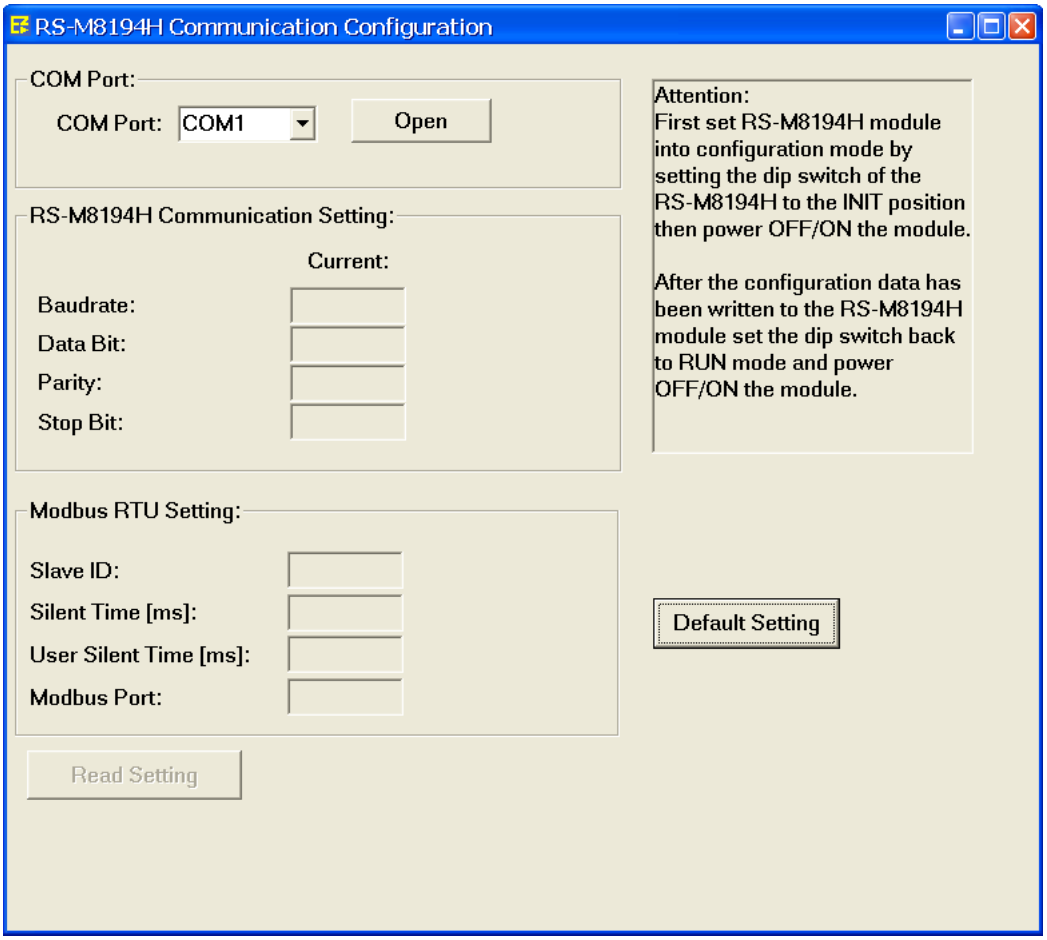


Figure 8: Open network configuration page

Step 7: Select the PC COM port to which the RS-M7184H is connect and then click the "***Open***" button.



Step 8: Click the "***Read Setting***" button to read the current serial configuration of

the device.

Step 9: Enter your new serial parameter setting.

Step 10: Click the "***Write New Setting***" button to download the parameter setting to the RS-M8194H. To restore the factory default setting just click the "***Default Setting***" and "***Write New Setting***" buttons.

Step 11: Set the DIP-switch back to the "Run" position and power off/on the RS-M8194H.

Step 12: IMPORTANT!!! The configuration RS232 port of the RS-M8194H is not isolated therefore it is necessary to remove the RS232 cable from this port after the setting to prevent electrical noises from affecting the device.

4 Command Types

More than 250 commands are being supported by the ET-M8194H\RS-M8194H. The commands can be separated into different categories:

1. RTC commands:
    - RTC commands are remote commands. These commands have to be sent by the remote Modbus master to directly control motion execution. The RTC commands differentiate two types of commands:
        - Interrupt commands
            - The commands are immediately being executed after arriving at the motion device, regardless of the current state of the device. Commands belonging to this type are stop commands and status commands.
        - Buffer commands
            - Motion commands arriving at the motion device are first written on a FIFO buffer and then are executed in sequence. A maximum of 29 commands can be stored on the buffer at a time.

2. MP commands
    - MP commands are Macro commands and can only be used inside a MP Macro program. A MP Macro program is a motion control program which is being saved to a nonvolatile memory inside the i8094H. A MP Macro program execution can be triggered by the Mobus master or an FRnet DI event.

3. ISR commands
    - ISR commands are intended for programming ISR Macro programs. The ISR Macro programs are being executed when a motion chip interrupt occurs. ISR Macro programs are very similar to MP Macro programs but in order to guarantee fast execution time only commands which can be executed in a very short period of time are allowed.

4. Initial Table (IT) command
    - Initial table commands include all the commands responsible for initializing the motion chip during power on. They are saved to the in volatile memory of the i8094H.

Function	RTC	MP	ISR	IT
SET_SLMT	◎	◎		◎
SERVO_ON	◎	◎		◎
MP_CALL	◎	◎		
MP_FOR		◎		
MP_IF		◎	◎	
SET_V	◎	◎	◎	
SET_A	◎	◎	◎	
FIXED_MOVE	◎	◎	◎	
CONTINUE_MOVE	◎	◎	◎	

Figure 9: Excerpt of the different command types

Most of the commands can either be used as RTC and as MP commands (Figure 9). All the configuration and settings options provided by the EzMove utility can also be done by directly calling the corresponding Modbus command for example via a HMI or by using the APIs provided by the ET_M8194H_API.dll\ RS_M8194H_API.dll. In this case the user has to write its own motion control program on the PC.

4.1 RTC commands

A RTC command has to be sent directly by the Modbus master. A Modbus master like the EzMove utility therefore has to be connected to the ET-M8194H \ RS-M8194H in order to call these commands. In contrast to Macro commands the RTC commands are not saved to the i8094H. As described in the following section most RTC commands are first cached before being executed.

The remote motion controller supports two kinds of RTC commands:

- Interrupt commands
- FIFO buffer commands

4.1.1 Interrupt commands

Interrupt commands are immediately being executed after arriving at the ET-M8194H and are not cached in the FIFO buffer. An interrupt command has got a high priority and interrupts the i8094H firmware. Therefore regardless of the ET-M8194H\ RS-M8194H and FIFO buffer state an interrupt command will always execute with the highest priority. Most interrupt commands are for reading the motion and IO status and for holding or stopping motion execution (Figure 10).

Command Type	Command Name
Position	– GET_LP – GET_EP
Velocity	– GET_CV
Acceleration	– GET_CA
Digital input	– GET_DI, GET_DI_ALL – FRNET_RA, FRNET_READ
Axis driving status	– GET_ERROR, GET_ERROR_CODE
Variable value	– READ_VAR, WRITE_VAR – READ_FLONG, WRITE_FLONG
Stop/ emergency stop	– STOP_SLOWLY, STOP_SUDDENLY, CLEAR_STOP, – VSTOP_SLOWLY, VSTOP_SUDDENLY, CLEAR_VSTOP, – EMERGENCY_STOP; CLEAR_EMERGENCY_STOP
Reset motion chip	– RESET_CARD

Figure 10: Interrupt commands

4.1.2 Buffer commands

Motion commands arriving at the ET-M8194H\ RS-M8194H are first written on to a FIFO buffer and then are executed in sequence. A maximum of 29 commands can be stored on the buffer at a time. Most RTC commands are cached.

The buffered commands will be added to the FIFO buffer of the ET-M8194H and once the command has been executed will be removed from the buffer. If the Master sends a RTC command while the buffer is full then the ET-M8194H\ RS-M8194H will respond with a Modbus exception code “SLAVE DEVICE BUSY” (Figure 11). The command therefore needs to be sent again. The FIFO buffer will be cleared automatically once an “EMERGENCY_STOP” or “RESET_CARD” command has been issued.

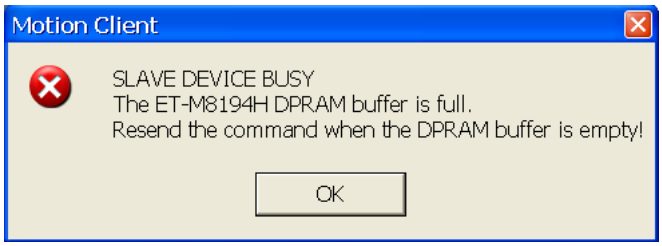


Figure 11: EzMove message box for indicating a Modbus exception code due to a full FIFO buffer

In some cases the next command will be read from the FIFO buffer even if the previous command has not finished. For example single axis commands like “FIXED_MOVE” can be overwritten by the following commands. To prevent this, a STOP_WAIT hast to be added between two executive “FIXED_MOVE” command to ensure that the first command has finished before starting the next command.

4.2 MP Macro commands

A MP Macro program is saved on the non volatile memory of the ET-M8194H\ RS-M8194H and its execution is initiated by a command sent by the Modbus master command or triggered through an FRnet DI event. The memory of the i8094H can save up to 157 MP Macro programs. A fixed memory size is reserved for each MP program as shown in Figure 12. The size indicates the maximum number of command lines a macro can hold. Some MP commands require more then one command line in order to be stored. The MP table number has to be selected according to the maximum number of commands lines required for the Macro program.

Macro Type	Macro Table Names	Table Qty	Command Lines
Motion Program (MP)	MP1 ~ MP40	40	8
	MP41 ~ MP90	50	16
	MP91 ~ MP130	40	32
	MP131 ~ MP150	20	64
	MP151 ~ MP155	5	128
	MP156 ~ MP157	2	512
Interrupt Service Routine (ISR)	ISR1 ~ ISR6	6	8
	ISR7 ~ ISR15	9	16
	ISR16 ~ ISR18	3	32
	ISR19 ~ ISR20	2	64

Figure 12: Fixed size Macro program table

MP commands are Macro commands and can only be used inside a MP Macro program. The start of a macro program is indicated by a ***MP_CREATE*** and the end with a ***MP_CLOSE*** command. Once a MP_CREATE command has been called the i8094H is in a macro download state and stays in this state until a MP_CLOSE command has been received. All the commands following the MP_CREATE command will be considered as MP commands and are saved in the receiving sequence to the i8094H. RTC buffer commands are being ignored by the module when it is in a macro download state.

The parameter of the MP_CREATE command indicates the Macro table name and therefore the size assigned for this Macro program (Figure 13).

```
MP_CREATE (MP Table No)

...
...
...

MP_CLOSE
```

```
MP_CALL (MP Table No)
```

Figure 13: Creating and calling a MP Macro program

MP commands are indicate by a “MP” in the command parameter interface followed by

the quantity of command lines required. For example “MP_FL x 1” (Figure 14) means that the command is a MP command which requires one command line for storing it in in the i8094H memory.

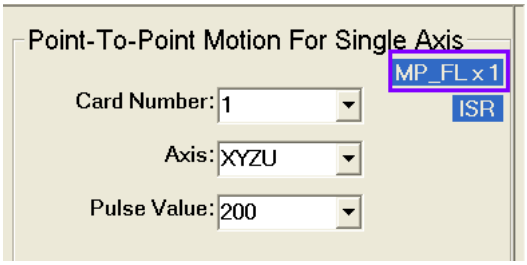


Figure 14: MP command

The main difference between a macro program and a batch program is shown in Figure 15:

MP/ISR Macro Program	Batch program (RTC)
• A Macro program always start with a ***MP_CREATE*** and ends with a ***MP_CLOSE*** command. The motion instruction entered between these two commands are part of the same macro. The same applies to the interrupt macros (ISRs) except they start with ***ISR_CREATE*** and end with ***ISR_CLOSE***. • Saved on a nonvolatile FRAM • Each macro program has a unique name. • A macro program can call another macro program. A macro can call an unlimited number of macros but in case of a nested macro call the number of nested layers is limited to 6 (Figure 16).	• Consist of sequence of motion instructions. No start and no end command. • Is written to a FIFO buffer. Can store a maximum off 29 commands at a time. • Commands will be executed immediately in the sequence they arrive at the buffer. After a command has been executed it will be removed from the buffer. • Batch editor allows the user to quickly write a set of motion commands

Figure 15: MP and batch program comparison

Example: Nested MP Call

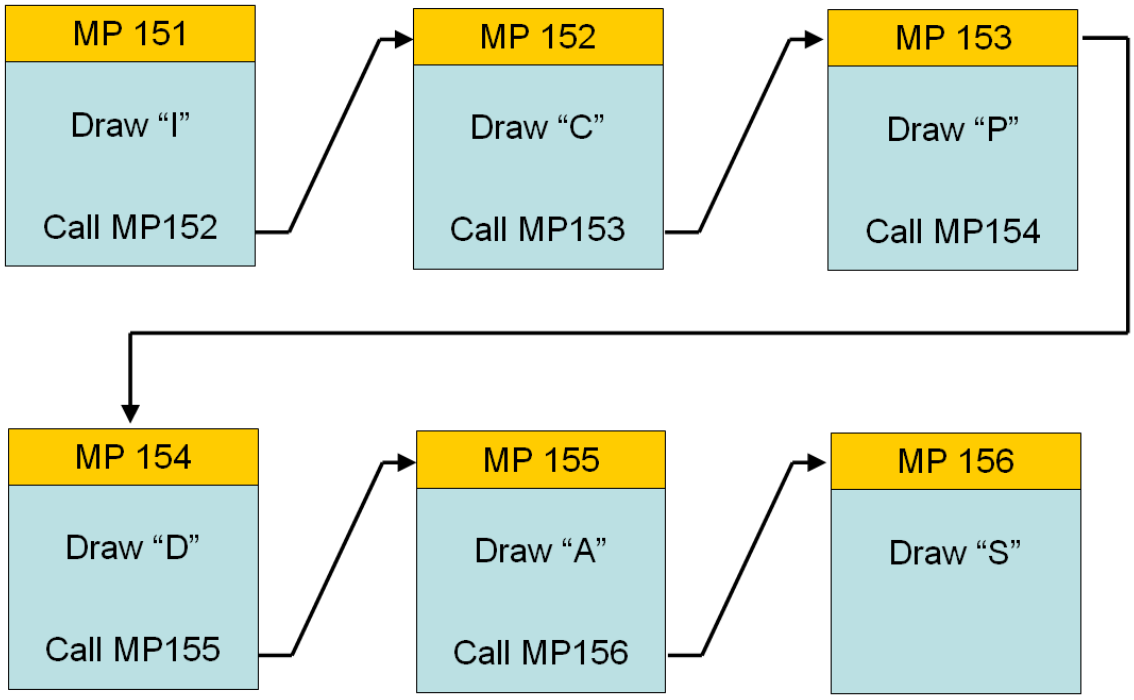


Figure 16: A 6 layer nested macro call

4.3 ISR Macro commands

An ISR Macro program execution is being triggered by a motion chip interrupt (Figure 17). The motion chip provides a number of interrupts for each axis which can be assigned to different ISR Macro programs. For each axis the following interrupt triggers are available:

- Pulse output
- End of driving
- Start / finish of a constant speed drive
- Position compare
  - Logical or encoder position counter is greater, smaller or equal to a set compare register value
- Automatic home search terminated
- Synchronous action was activated

Twenty ISR Macro programs can be store on the i8094H. Figure 12 shows the maximum quantity commands each ISR program can accommodate. In order to guarantee that ISR Macros are executed in a very short period of time the maximum Macro length is limited to 64 command lines, in addition no looping or delay commands are allowed. Most ISR commands are identical to the MP Macro commands.

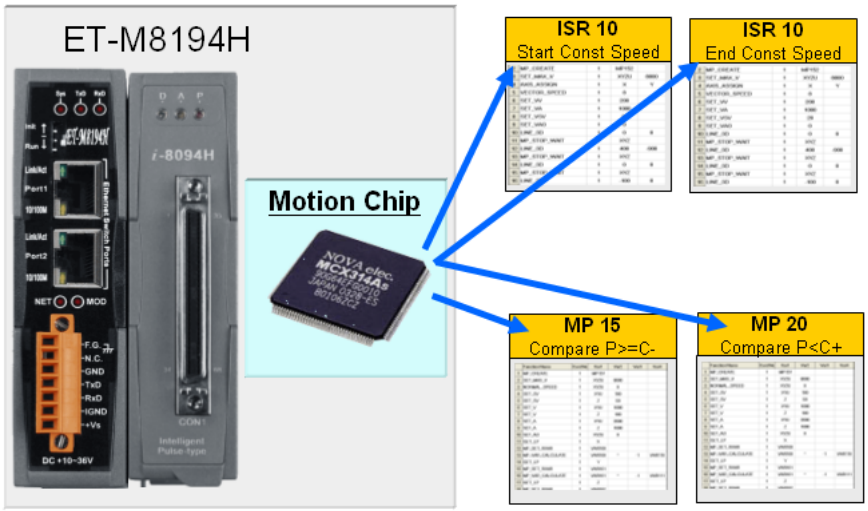


Figure 17: Motion chip interrupt triggered ISR Macro execution

A ISR Macro program is created in the same way as a MP Macro except that a ISR Macro program always start with a ***ISR_CREATE*** and ends with a ***ISR_CLOSE*** command (Figure 18). The Master can execute a ISR Macro program by sending the MP_ISR_CALL command. It is recommended to just use this command for testing the ISR program. In general ISR programs should only be triggered by the motion chip interrupt.

```
MP_ISR_CREATE ( ISR Table No)

...
...
...

MP_ISR_CLOSE
```

Figure 18: ISR Macro program definition

4.3.1 Procedure to enable ISR Macro program execution

The motion chip can only interrupt the i8094H when the interrupt has been enabled. The command **ENABLE_INT** enables the chip interrupt. Now the selected chip interrupt factor (interrupt trigger) has to be enabled and assigned to an ISR Macro program by calling **INTFACTOR_ENABLE**. Both of these commands can be called inside a MP Macro program. Once the interrupt with the specified interrupt factor occurs the interrupt factor is automatically being disabled. Therefore if the interrupt factor needs to be active after an interrupt occurs the **INTFACTOR_ENABLE** has to be called inside the ISR program to enable the same interrupt factor again.

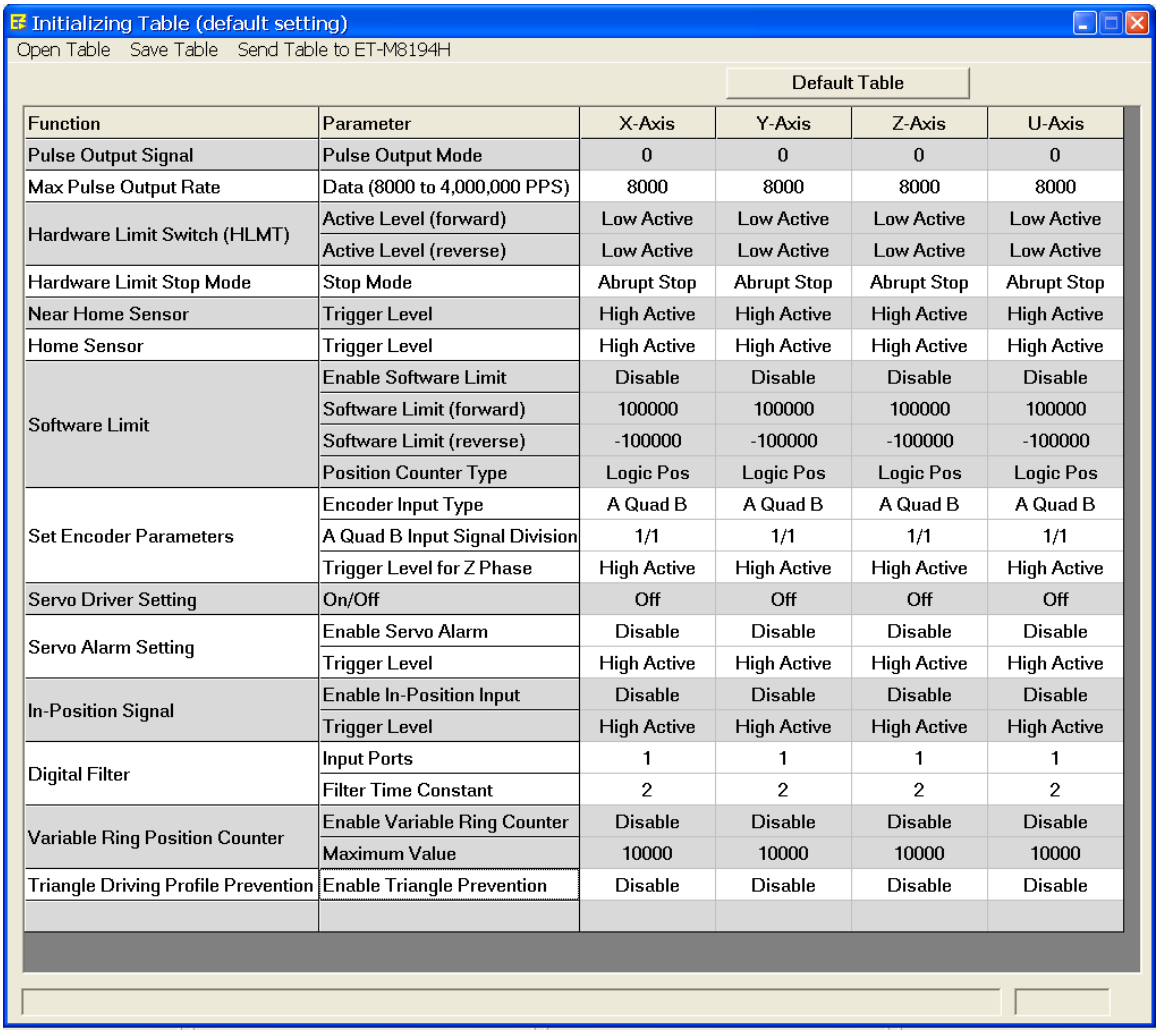
Procedure to enable an interrupt:

1. Assign the interrupt macro to a interrupt factor
   ```
   INTFACTOR_ENABLE(1, BYTE axis, BYTE nINT, BYTE ISRNo)
   ```
2. call the function
   ```
   ENABLE_INT(1)
   ```

The interrupt macro will be called when the assign interrupt occurs.

4.4 Initial Table (IT) commands

The Initial Table includes all the commands which are required to initialize the motion chip after power on. It is important that the motion chip setting correspondents to the servo drive setting otherwise the system will not be able to function properly. The Initial Table interface provides an easy and convenient way to enter the initial parameters.



Function	Parameter	X-Axis	Y-Axis	Z-Axis	U-Axis
Pulse Output Signal	Pulse Output Mode	0	0	0	0
Max Pulse Output Rate	Data (8000 to 4,000,000 PPS)	8000	8000	8000	8000
Hardware Limit Switch (HLMT)	Active Level (forward)	Low Active	Low Active	Low Active	Low Active
	Active Level (reverse)	Low Active	Low Active	Low Active	Low Active
Hardware Limit Stop Mode	Stop Mode	Abrupt Stop	Abrupt Stop	Abrupt Stop	Abrupt Stop
Near Home Sensor	Trigger Level	High Active	High Active	High Active	High Active
Home Sensor	Trigger Level	High Active	High Active	High Active	High Active
Software Limit	Enable Software Limit	Disable	Disable	Disable	Disable
	Software Limit (forward)	100000	100000	100000	100000
	Software Limit (reverse)	-100000	-100000	-100000	-100000
	Position Counter Type	Logic Pos	Logic Pos	Logic Pos	Logic Pos
Set Encoder Parameters	Encoder Input Type	A Quad B	A Quad B	A Quad B	A Quad B
	A Quad B Input Signal Division	1/1	1/1	1/1	1/1
	Trigger Level for Z Phase	High Active	High Active	High Active	High Active
Servo Driver Setting	On/Off	Off	Off	Off	Off
Servo Alarm Setting	Enable Servo Alarm	Disable	Disable	Disable	Disable
	Trigger Level	High Active	High Active	High Active	High Active
In-Position Signal	Enable In-Position Input	Disable	Disable	Disable	Disable
	Trigger Level	High Active	High Active	High Active	High Active
Digital Filter	Input Ports	1	1	1	1
	Filter Time Constant	2	2	2	2
Variable Ring Position Counter	Enable Variable Ring Counter	Disable	Disable	Disable	Disable
	Maximum Value	10000	10000	10000	10000
Triangle Driving Profile Prevention	Enable Triangle Prevention	Disable	Disable	Disable	Disable

Figure 19: Initial Table interface

The initial setting of the motion chip can be altered at any time during runtime by calling the commands listed in Figure 20. From Figure 20 it is apparent that all IT commands can be called inside a MP Macro program or directly called by the Modbus master (RTC). The IT commands are described in detail in the ET-M8194H\ RS-M8194H Manual.

Function	RTC	MP	ISR	IT
SET_PULSE_MODE	◎	◎		◎
SET_MAX_V	◎	◎		◎
SET_HLMT	◎	◎		◎
LIMITSTOP_MODE	◎	◎		◎
SET_NHOME	◎	◎		◎
SET_HOME_EDGE	◎	◎		◎

SET_SLMT	◎	◎		◎
CLEAR_SLMT	◎	◎		◎
SET_ENCODER	◎	◎		◎
SERVO_ON	◎	◎		◎
SERVO_OFF	◎	◎		◎
SET_ALARM	◎	◎		◎
SET_INPOS	◎	◎		◎
SET_FILTER	◎	◎		◎
VRING_ENABLE	◎	◎		◎
VRING_DISABLE	◎	◎		◎
AVTRI_ENABLE	◎	◎		◎
AVTRI_DISABLE	◎	◎		◎

Figure 20: Initial Table commands

5 Editing a Macro Program

EzMove utility assists the user in programming, debugging and testing simple motion control Macros. It can only be used together with the ET-M-8194H\ RS-M8194H motion control module.
The motion Macro commands have to be selected from the command list. This prevents the user from editing wrong command names or editing wrong parameters. With EzMove you can create two different Macro program types:

- Macro program (MP): This Macro contains the actual motion control command
- Interrupt service routine Macros program (ISR): Is being called when an interrupt of the motion chip occurs.

The macro programs have to be downloaded to the ET-M-8194H device. Macros are stored in a nonvolatile memory.

The ET-M-8194H \ RS-M8194H device supports up to 157 Macro tables (MP1~MP157) with different sizes (stacks). The size indicates how many command lines a Macro table supports. Six categories of fixed size Macro tables are provided (Figure 12: 8/16/32/64/128/512 stacks). A Macro table has to be selected according to the number of motion commands to be used inside a Macro program. A Macro form with a table size of 32 has space for up to 32 commands.

A macro program always start with a ***MP_CREATE*** and ends with a ***MP_CLOSE*** command.

Note:

- Always make sure to call MP_CLOSE at the end of a macro otherwise all the commands following the Macro will be automatically included to the Macro untill the memory reserved for the macro is full.
- Make sure that not more commands are assigned to a Macro tables than it can hold. For example the Macro MP1 reserves memory for eight command lines. That means the Macro program can hold between the *MP_CREATE* and *MP_CLOSE* up to seven commands. If more then seven instructions are downloaded the remaining commands are ignored and get lost. *MP_CREATE* is not a Macro command and therefore is not written to the Macro table.
- Remember that some commands require more space in memory than reserved for one command line. For example the ARC_CW and ARC_CCW each require memory space for two command lines. This reduces the number of commands a Macro can hold. Consult the *ET_M8194H_Manual* \

RS_M8194H_Manual or click ***Help ->FLine Table*** to look up the number of command lines a command requires.
In addition every parameter input interface indicates how many lines a command requires:

MP_FLx1 – one line
MP_FLx2 – two lines
MP_FLx3 – three lines
MP_FLx4 – four lines

The following section describes how to

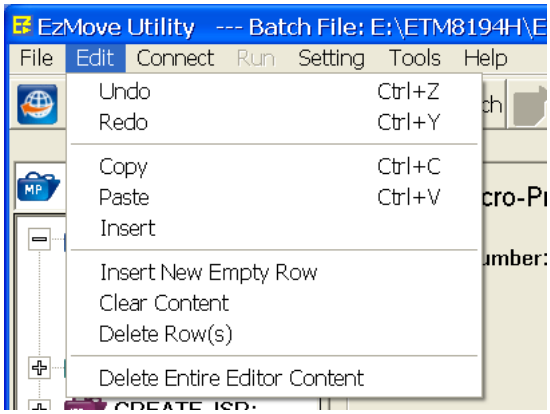
- program a macro for doing a rectangular motion in a X-Y plane,
- download the macro program to the nonvolatile memory of the ET-M8194H\ RS-M8194H device and
- execute the macro program.

5.1 Macro Editor

The Macro editor is a simple interface for writing Macro programs. Commands can be directly entered in the editor. For each command parameter a combo box provides a number of valid selections. Conditional instruction, loop and jump statement supports the writing of a motion control program. Commands which have been spelled incorrectly or are not supported are indicated by a red background color. Normal text editing functions like copy, cut, paste, undo, redo, delete, etc. are supported. An alternative to using the editor to write Macro programs is to use Excel. Excel files can be opened by the Macro editor and then send to the ET-M8194H\ RS-M8194H.

5.1.1 Basic features

In the following the basic text editing functions supported by the utility is being introduced.



- Undo and redo
  You can undo and redo up to 20 actions. To undo an action, do one or more of the following:
  - Click Undo on the EzMove toolbar.
  - Keyboard shortcut CTRL+Z.

  To redo an action that you undid,
  - click Redo on the EzMove toolbar or
  - use the keyboard shortcut CTRL+Y.

- Copy and paste
  Single or multiple lines in the editor can be copied and paste to any other line. Only whole command lines can be copied. In addition command lines can be copied from the editor into an Excel table and vice versa. To select lines just left click the start line and scroll to the end line then right click the moue to select the desired action from the popup menu or use the keyboard short cuts.

- Insert and delete rows
  Copied rows or empty rows can be inserted between existing rows. The content of marked rows can be deleted or the entire row with its contend can be deleted.

5.1.2 Macro editing method 1: Using command parameter interface

A command first has to be selected from the command list, and then its parameters have to be set and thereafter it has to be added to the Macro editor.

Step 1: The motion commands are arranged into 11 categories to facilitate the selection. Select "***Macro Program Functions***" category.

Step 2: Click the ***MP_CREATE*** command. Every MP Macro has to start with the command ***MP_CREATE.***

Step 3: Select ***MP93*** as macro form. This form provides lines for up to 32 commands.

Step 4: Click the "Add to Batch" button to add the new command to the ***Macro/Batch*** editor

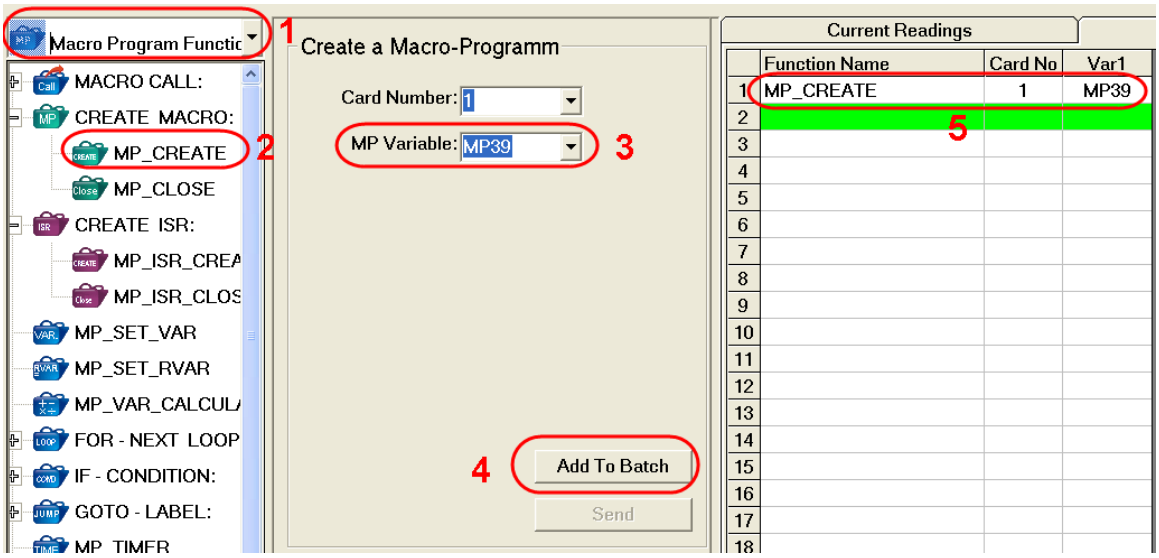


Figure 21: Editing commands (method 1)

5.1.3 Macro editing method 2: Using Macro program editor

Command and their parameter can be directly selected from a dropdown list in the Macro/Batch editor.

Step 1: Click the line directly beneath the MP_CREATE command of the *first* column of the Macro/Batch editor. A dropdown list with all the commands supported by the ET-7184H\ RS-M8194H device pops up. Select ***SET_MAX_V***.

Step 2: Now click the column of the first variable (Var1). Select ***XYZU***. This means the velocity resolution which you are going to set applies to all axes.

Step 3: Click the column of the second variable (Var2). Enter a velocity resolution value of ***8000***.

Step 4: Click the first column of the next line to select the next command.

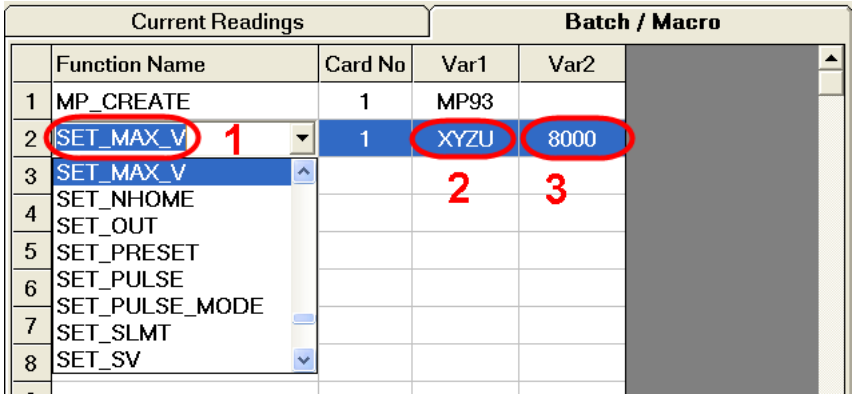


Figure 22: Editing commands (method 2)

Step 5: Continue entering the following Macro commands either using method 1 or 2.

	Function Name	Card No	Var1	Var2
1	MP_CREATE	1	MP93	
2	SET_MAX_V	1	XYZU	8000
3	NORMAL_SPEED	1	XYZU	0
4	SET_V	1	XYZ	200
5	SET_A	1	XYZ	1000
6	SET_SV	1	XYZ	20
7	SET_AO	1	XYZ	0
8	SET_LP	1	XYZU	0
9	FIXED_MOVE	1	Z	100
10	MP_STOP_WAIT	1	Z	
11	MP_TIMER	1	2000	
12	FIXED_MOVE	1	XY	100
13	MP_STOP_WAIT	1	XY	

14	FIXED_MOVE	1	Z	-100
15	MP_STOP_WAIT	1	Z	
16	FIXED_MOVE	1	Y	800
17	MP_STOP_WAIT	1	Y	
18	FIXED_MOVE	1	X	800
19	MP_STOP_WAIT	1	X	
20	FIXED_MOVE	1	Y	-800
21	MP_STOP_WAIT	1	Y	
22	FIXED_MOVE	1	X	-800
23	MP_STOP_WAIT	1	X	
24	MP_CLOSE	1		

This macro has already been created by ICPDAS and has been saved as ***MP93_Square.dat*** to the following directory:
C:\ICPDAS\ET_M8194H\EzMove_Utility\Demo
Open this file with the EzMove utility by clicking the "Open Macro / Batch File" button

Step 6: Save the Macro program to file: Click "Save as Macro / Batch File" and enter a name for the file. The file extension is ***.dat***.

5.2 Sending single commands

Once a batch or macro has been created individual commands can be selected from the editor and send to the ET-M8194H device for immediate execution. This feature is only supported by commands which name do not start with "MP". This feature enables the programmer to debug his batch/macro by sending the commands one by one.

Example:

Open the ***MP93_Square.dat*** in the directory
C:\ICPDAS\ET_M8194H\EzMove_Utility\Demo

The following steps refer to Figure 23:

Step 1: Click ***SET_MAX_V*** and then "***Send***".
Step 2: Click ***NORMAL_SPEED*** and then "***Send***".
Step 3: Click ***SET_V*** and then "***Send***".

After a motion command like FIXED_MOVE has been sent you can wait until the command has been executed and check whether the motion system has really reached its target position. Then you can dispatch the next motion command.

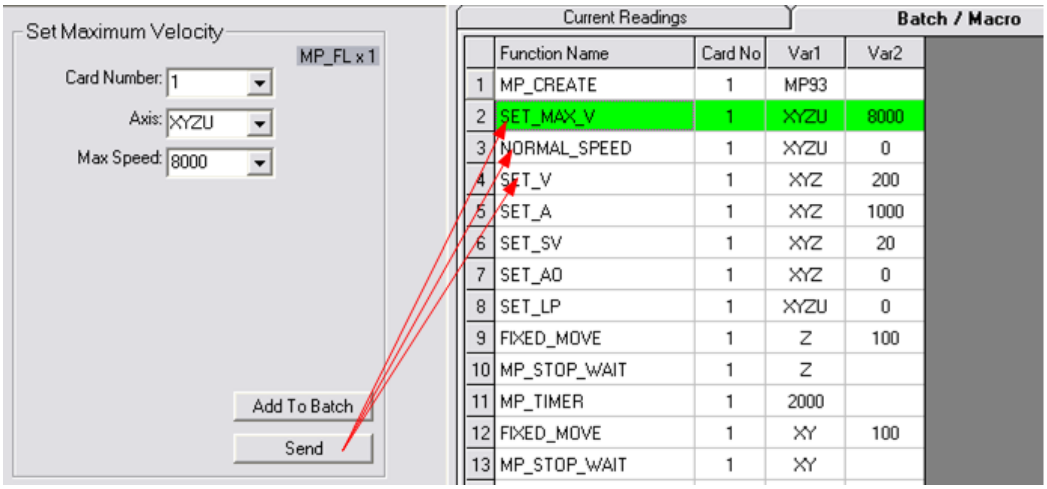


Figure 23: Sending and executing commands one by one

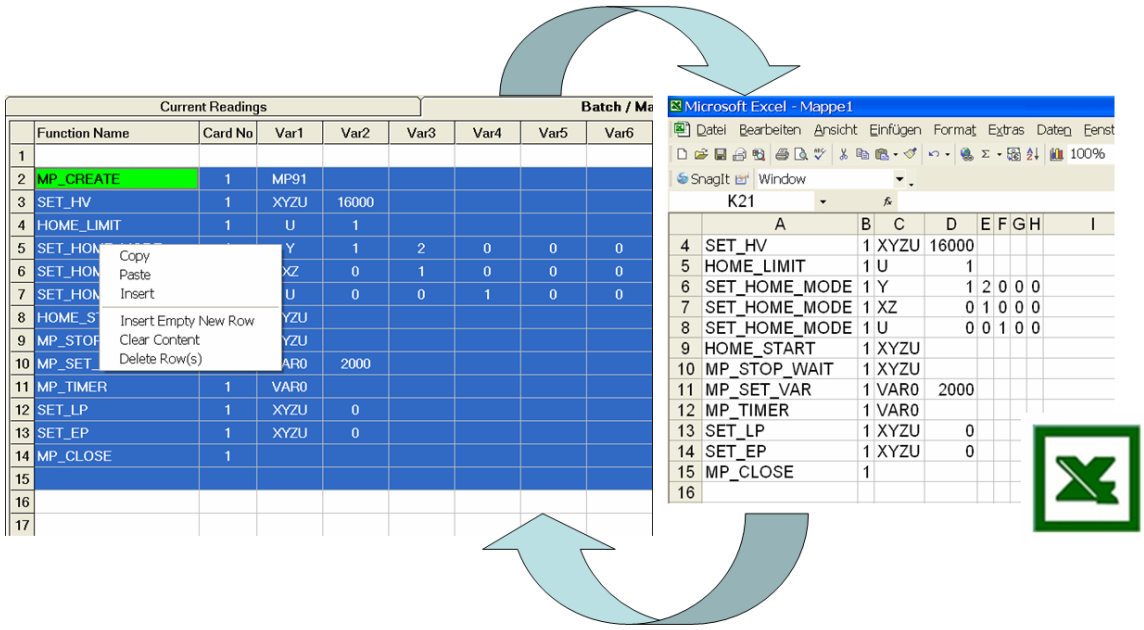
5.3 Writing Macro program using Excel

Macro program edited in the Macro\Batch editor can be saved as an Excel file. The programmer can use the Excel file to modify the Macro program. When editing commands to the Excel file make sure that the command name and the number of parameters are correct.

	A	B	C	D
1	Function Name	Card No	Var1	Var2
2	MP_CREATE	1	MP93	
3	SET_MAX_V	1	XYZU	8000
4	NORMAL_SPEED	1	XYZU	0
5	SET_V	1	XYZ	200
6	SET_A	1	XYZ	1000
7	SET_SV	1	XYZ	20
8	SET_AO	1	XYZ	0
9	SET_LP	1	XYZU	0
10	FIXED_MOVE	1	Z	100
11	MP_STOP_WAIT	1	Z	
12	MP_TIMER	1	2000	
13	FIXED_MOVE	1	XY	100
14	MP_STOP_WAIT	1	XY	
15	FIXED_MOVE	1	Z	-100
16	MP_STOP_WAIT	1	Z	
17	FIXED_MOVE	1	Y	800
18	MP_STOP_WAIT	1	Y	
19	FIXED_MOVE	1	X	800
20	MP_STOP_WAIT	1	X	
21	FIXED_MOVE	1	Y	-800
22	MP_STOP_WAIT	1	Y	
23	FIXED_MOVE	1	X	-800
24	MP_STOP_WAIT	1	X	
25	MP_CLOSE	1		

Figure 24: Macro program edited via Excel

EzMove utility allows the user to directly copy data from the Macro editor to Excel and vise versa.



EzMove utility allows the user to directly copy data from the Macro editor to Excel and vise versa.

Figure 25: Copy and paste data between EzMove and Excel

5.4 Macro program download

The procedure for downloading Macro programs to the ET-M8194H\ RS-M8194H is as follows:

Step 1: Power the ET-M8194H\ RS-M8194H device on.
Make sure the dip switch is set to "Run" before you power on the device.

Step 2: Connect the EzMove utility to the remote motion device.
Open the "Connect" dialog window (click "***Connect→Connect to Remote Device…***")

- For the ET-M8194H enter the IP address, connection timeout and Modbus timeout; then click "***Connect***" (Figure 26). The status bar displays a green icon when the utility is connected to the ET-M8194H device.

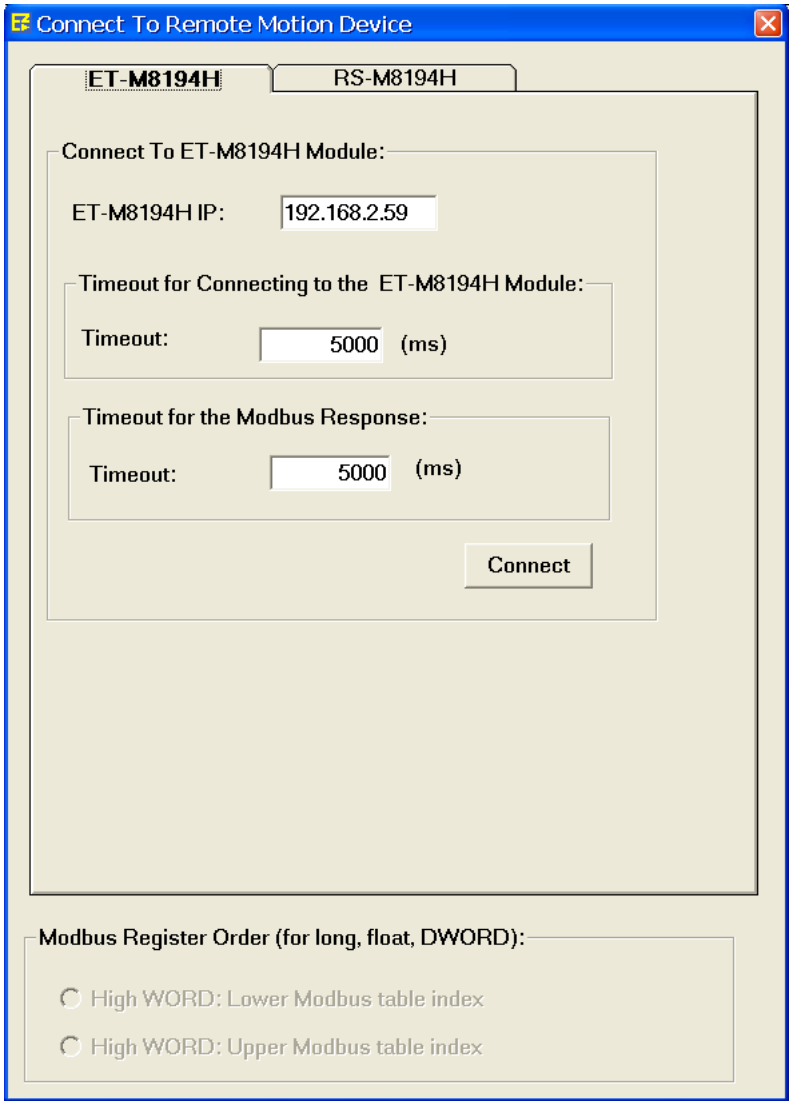


Figure 26: ET-M8194H connection interface

- For the RS-M8194H enter the serial COM port parameters, slave ID and response timeout, then click "***Connect***" (Figure 27). The status bar displays a green icon when the utility is connected to the RS-M8194H device.

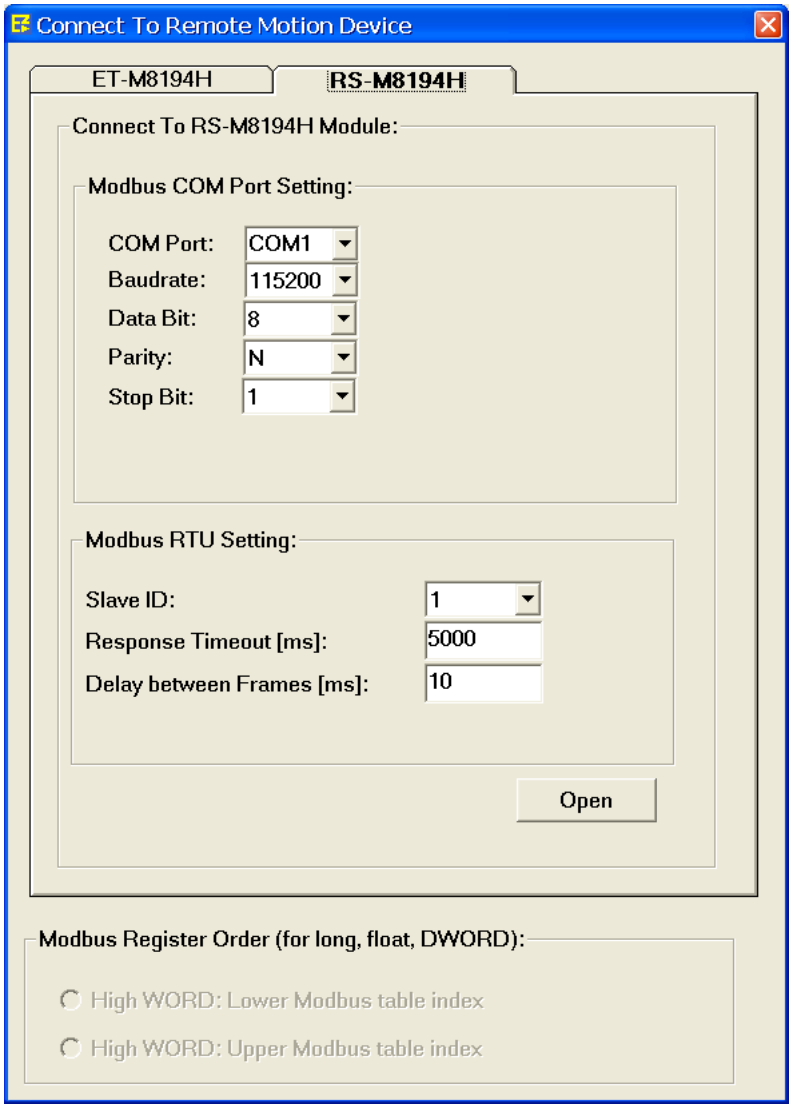


Figure 27: RS-M8194H connection setting

Step 3: Click the “Download” button in the toolbar.

The macro “***MP93***” is now being downloaded to a nonvolatile memory.

Note: To have a clear structure and to make debugging easier it is suggested to edit and download only one macro at a time. Save each Macro program created with the EzMove to a different file and download each Macro program seperatly.

5.5 Executing Macro programs

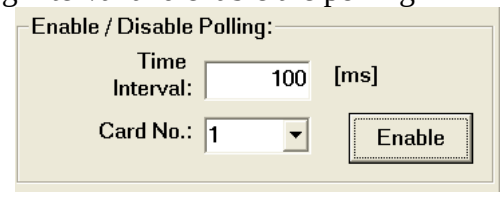
There are three methods to trigger a Macro program execution:

- The Modbus master directly calls a Macro program
- The motion chip executes a Macro program via interrupt
- A FRnet DI event calls a designated Macro program

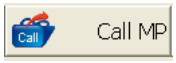
5.5.1 Macro Program execution via Modbus Master

The EzMove utility acts as a Modbus master. The following steps describe the procedure to execute a Macro program:

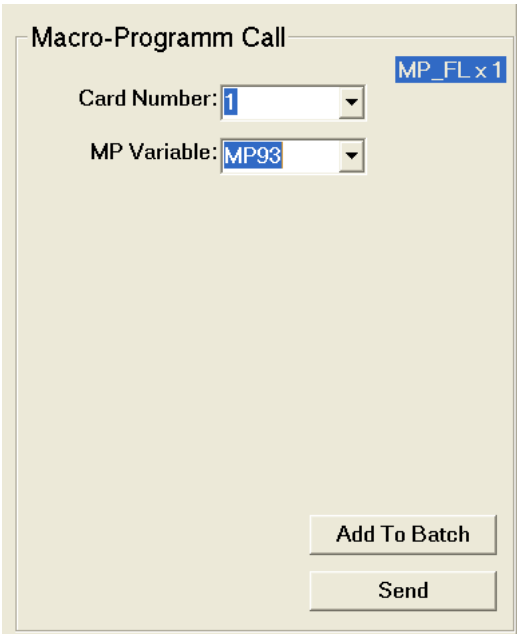
Step 1: Set the polling interval and enable the polling



Step 2: Click the “Call MP” button in the toolbar.



Step 3: Select the Macro program number you like to execute. In this case it is program “MP93”



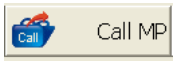
Step 4: Click “Send” to run the Macro program. The program will now be executed.

5.5.1.1 Calling multiple Macro programs

It is also possible to send multiple Macro program calls to the ET-M8194H\ RS-M8194H. The program calls will be saved to the FIFO buffer while a Macro program is busy executing. Once a Macro program has ended the next program call will be read from the FIFO buffer. Up to 29 program calls can be stored to the buffer.

The calling procedure is as follows:

Step 1: Click the “Call MP” button in the toolbar.



Step 2: Select a Macro program number you like to execute then click “*Add to Batch*”. In this case the program is not being called immediately but added to the Batch/Macro editor. Now add the next Macro program call to the editor. In Figure 28 seven Macro calls are added to the editor.

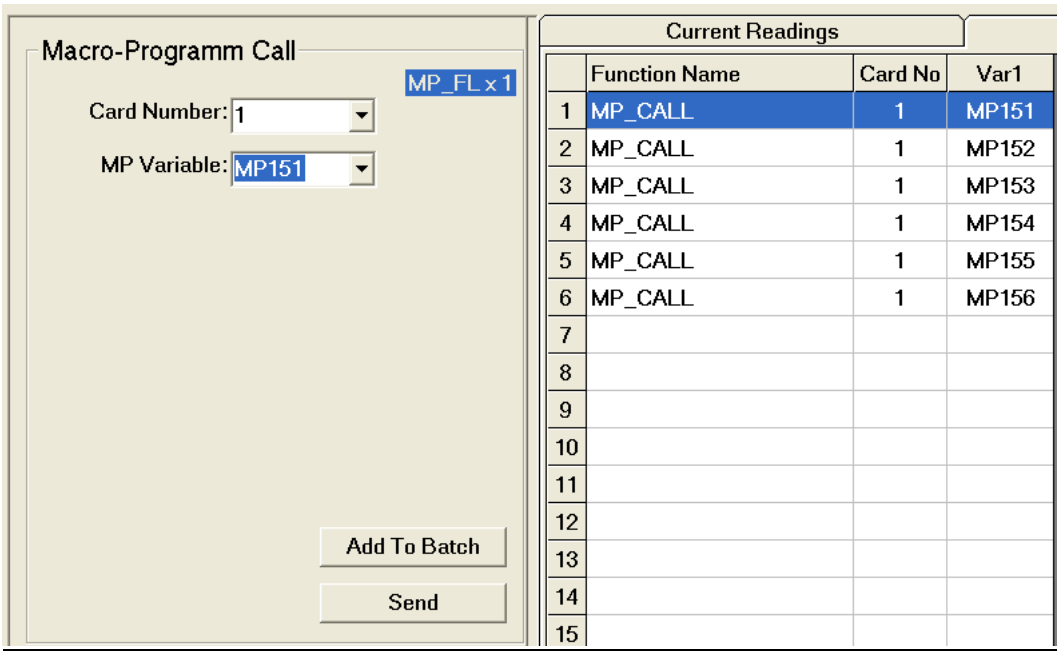
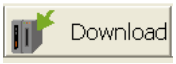


Figure 28: Multiple Macro program calls

Step 3: Click the “Download” button in the toolbar.



The macros will now be executed one by one in the same order as they are listed in the batch/macro editor.

Step 2 and Step 3 can be bypassed by directly clicking the “Send” button (Figure 28). The Macro program call will be directly send to the FIFO buffer.

5.5.2 Macro Program execution via FRnet Di

The procedure for using FRnet DI to execute a Macro program is described in chapter 6.

5.5.3 Emergency stop

The emergency stop command stops all motion execution immediately. In addition the current running Macro program will be terminated and all in the FIFO buffer existing

commands will be deleted. The ET-M8194H\ RS-M8194H stays in the emergency stop mode until the mode has been cleared by “Clear Emergency Stop”. The emergency stop state is being indicated by the red flashing “Clear Emergency Stop” button.

Figure 29: Emergency stop related buttons

5.6 Displaying motion pathway

EzMove allows the user to display the logical or encoder path while the ET-M8194H\ RS-M8194H is executing motion commands.

Step 1: Click “***Tools ->Graph***” in the menu bar. A graph window with a three axis coordinate system pops up.

Step 2: Enter the range for each axis (units are pulses) then click “***Set Axis Range***”.

Step 3: Select whether to display the logical or encoder position.

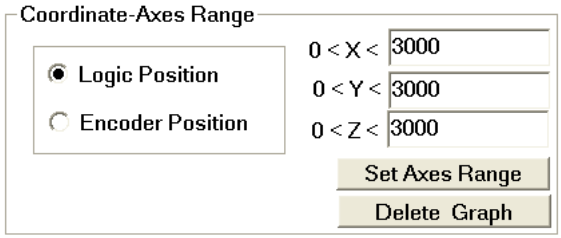
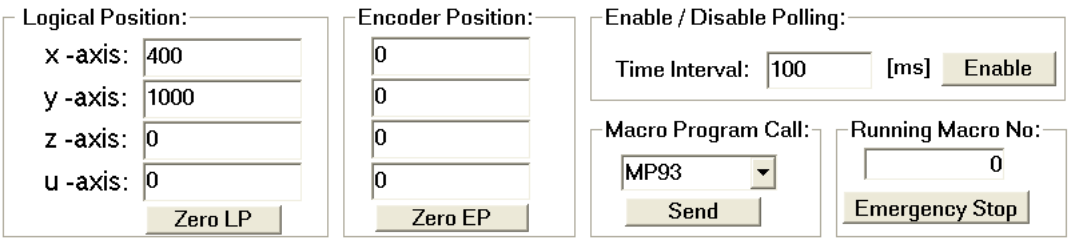


Figure 30: Axes range

Step 4: Click the “***Current Readings***” tab. Enter 100 milliseconds for the polling time interval and click “***Enable***”. The EzMove will poll every 100 milliseconds the logical and encoder position.



Step 5: Set both the logical and encoder position to zero by clicking the "***Zero LP***" and "***Zero EP***" buttons.

Now call and execute a Macro program:

Step 6: Select a Macro program number to execute from the combo box.
In this example Macro program "MP93" will be called as this is the only macro we have so far downloaded.

Step 7: Click "Send" to run the macro. The macro will now be executed.

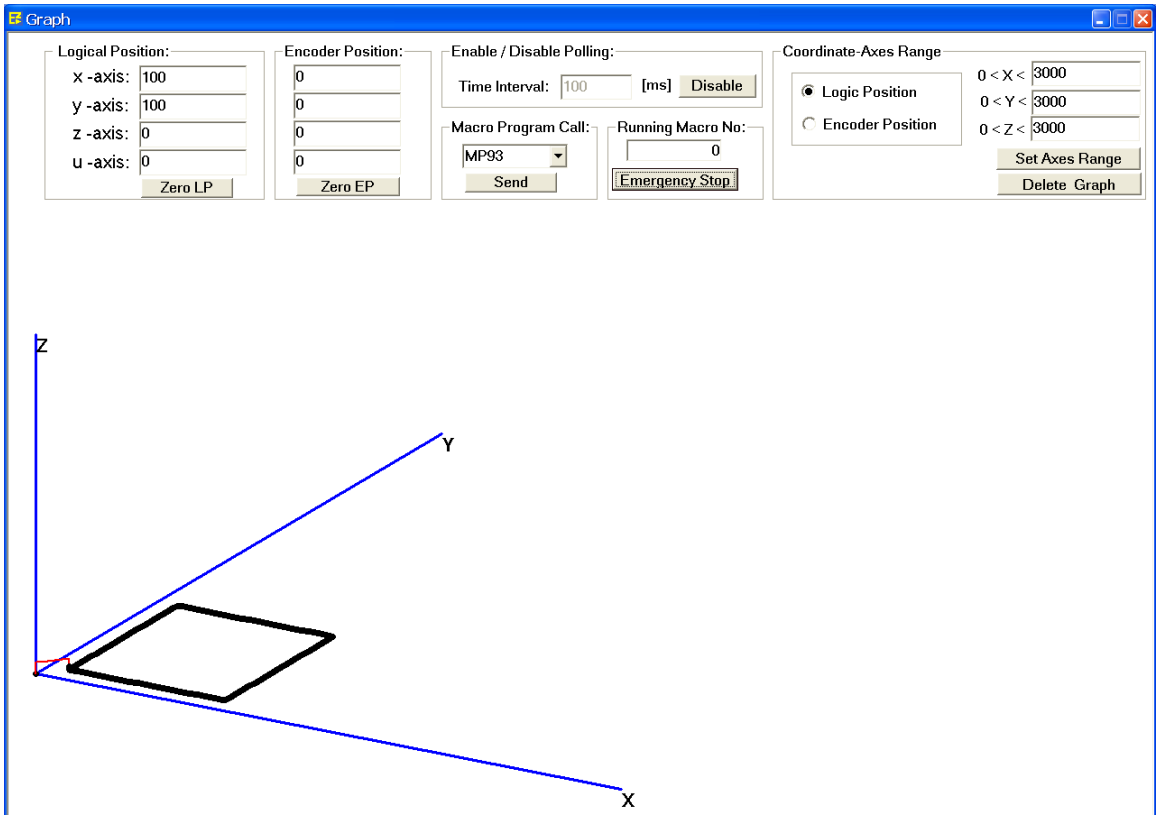


Figure 31: Rectangle of MP93

6 FRnet DI Event

The ET-M8194H\ RS-M8194H acts as a FRnet master and can control up to 128 digital outputs and 128 digital inputs. FRnet is a two-wire serial bus and has a scan interval of 2.88 ms and it is specifically designed for easy and cost effective wiring. ICPDAS provides a large range of FRnet I/O modules and terminal boards. ICPDAS provides two types of FRnet modules: digital input and digital output devices with 16 channels.

Up to 16 FRnet slaves can be connected to each master port: a maximum of 8 digital input and 8 digital output FRnet slaves. Every FRnet slave has got 16 channels. Each slave module is identified by its address. The first eight slave addresses (00 to 07) are reserved for digital output modules and the remaining eight addresses (08 to 15) are reserved for digital input modules. The master therefore can control via one port up to 128 digital output and 128 digital inputs. Knowledge of the FRnet protocol is not required as the provided commands access the FRnet modules (Figure 32).

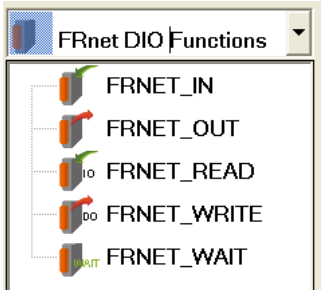


Figure 32: FRnet commands

The DI channels of the connected FRnet module can be set to trigger certain events. The events provided are as follows:

- Macro program call: Any Macro program number (MP1 to MP157) can be assigned to an FRnet DI event. It is important to note that a FRnet initiated program call does not interrupt a currently running Macro program. The ET-M8194H will first finish the current program before starting executing the program call generated by the FRnet DI event. Therefore this is not identical to an interrupt ISR Macro program call.
- Hold a single axis command: The hold command prevents the next motion command from being executed but does not prevent the current running motion command from continue executing until it has finished. If the trigger condition for the hold command is “OFF to ON” then the “ON to OFF” event of the same DI channel will automatically release the hold command and the motion will continue with the command which has been put to a hold.
- Abort current command: The currently running command will be aborted once

the corresponding event has been triggered and the program will continue executing the next command.

- Abort Macro program: Once the event is generated the current Macro program is being aborted but the current executing motion command will still continue executing until it has finished.
- Pause command: The pause command is very similar to the hold command. The difference is that the hold command only holds the execution of the next motion command in the Macro program whereas the pause command stops any next command in the Macro program from being executed until the pause mode has been released. Only one DI of all the FRnet modules can be assigned to a pause event.

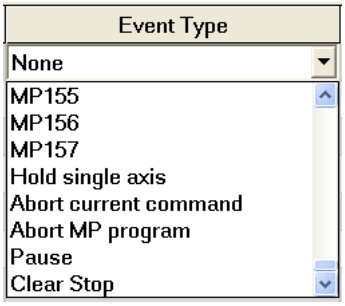


Figure 33: FRnet event types

The group number (Figure 34) represents the address of the different FRnet modules. The FRnet DI starts with the group number 8 to 15. The channel number represents the DI channel of the corresponding module. The events assigned to the DI channels have got a certain execution priority order:

- DI events generated by a module with a lower group number have got a higher priority then events generated by a module with a higher group number.
- Event triggered by a DI channel with a lower channel number will be executed before a DI event from a higher channel number.

6.1 Event Configuration

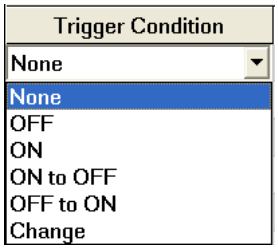
FRnet DI		Setting		
Group	Channel	Trigger Condition	Event Type	Axis
8	0	OFF to ON	MP1	
	1	ON to OFF	MP18	
	2	Change	MP137	
	3	Change	Abort Command	
	4	OFF to ON	Abort Macro Program	
	5	OFF to ON	MP35	
	6	ON	Hold Drive	X
	7	ON	Hold Drive	Y
	8	ON	Hold Drive	Z
	9	ON	Hold Drive	U
	10	ON	Hold Drive	XYZU
	11	None	None	
	12	None	None	
	13	None	None	
	14	None	None	
	15	None	None	
9	0	None	None	
	1	None	None	
	2	None	None	
	3	None	None	
	4	None	None	
	5	None	None	
	6	None	None	

Figure 34: FRnet Di event configuration interface

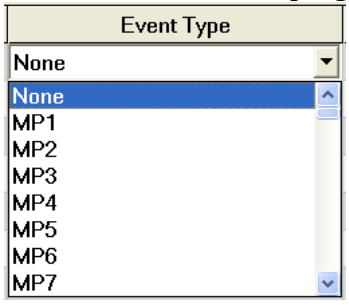
To assign an FRnet DI channel event to a Macro program number proceeds as follows:

Step 1: Open the FRnet DI event setting window:
"***Setting -> FRnet DI***"

Step 2: Select for a channel which acts as a Macro program trigger the trigger condition from the combo box list.

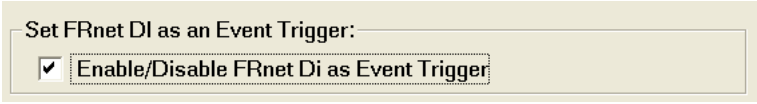


Step 3: Select the number of the Macro program to execute



Step 4: Download the setting to the remote motion controller

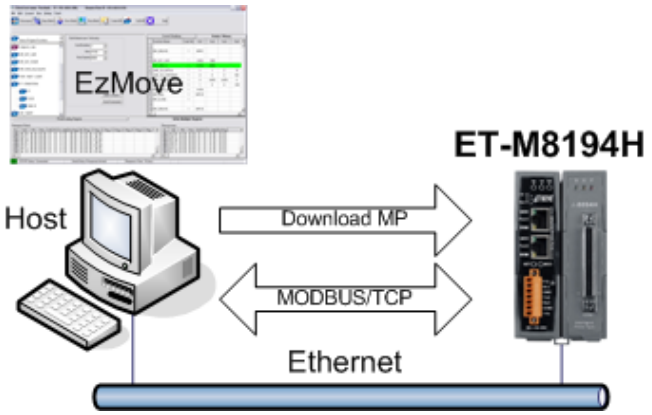
Step 5: Enable the FRnet DI as a event trigger by checking the check box:



FRnet is no ready to be used as a Macro program event trigger.

7 Modbus Communication

The communication protocol between the master (PC) and the ET-M8194H\RS-M8194H slave device is Modbus. It is important to note that according to the Modbus protocol only the master can initiate a transaction and the slaves only respond to a request of the master. A slave never initiates a transaction and always responds to or confirms a master request.

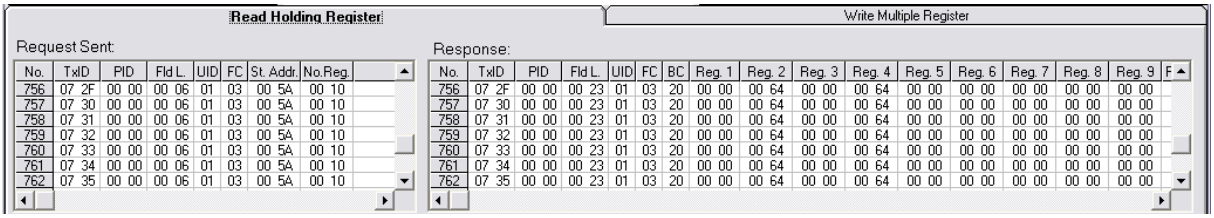


7.1 Modbus message structure

The tab window at the bottom of the EzMove utility shows for each dispatched command the actual data field send to the ET-M8194H\RS-M8194H device and the corresponding response data field.

7.1.1 Reading holding register

This Modbus data format is being used for polling data such as reading velocity, acceleration, logical and encoder position.



The data field of the following Table 1 reads the logical and encoder position of the four axes. As each position is stored in two registers altogether 16 register have to be read.

Request			Response		
Field Name	Length (Bytes)	(Hex) Example	Field Name	Length (Bytes)	(Hex) Example
Transaction ID:	2	07 2F	Transaction ID:	2	07 2F
Protocol ID:	2	00 00	Protocol ID:	2	00 00
Field Length:	2	00 06	Field Length:	2	00 23
Unit ID	1	01 *(cardNo)*	Unit ID	1	01 *(cardNo)*
Function Code:	1	03	Function Code:	1	03
Address of first input Register:	2	00 5A *(90)*	Byte Count:	1	20
Number of Register:	2	00 10 *(16)*	Input Register (00 5A):	2	00 00 *(MSW LP_X)*
			Input Register (00 5B):	2	00 64 *(LSW LP_X)*
			Input Register (00 5C):	2	00 00 *(MSW LP_Y)*
			Input Register (00 5D):	2	00 64 *(LSW LP_Y)*
			…	2	….
			Input Register (00 69):	2	…

MSW- most significant word
LSW – least significant word

Table 1: Reading logical and encoder position

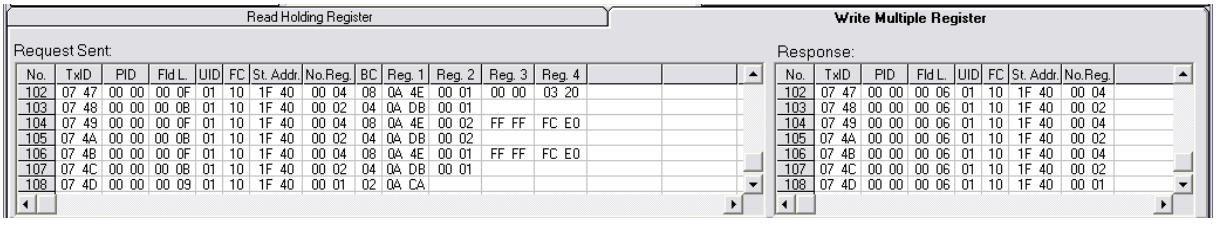
Function	Holding Register	Description

	Address	
LP_X	90 (0x005A)	Logical position of X-axis. It takes two registers. The MSW is located at address 90.
LP_Y	92 (0x005C)	Logical position of Y-axis. It takes two registers. The MSW is located at address 92.
LP_Z	94 (0x005E)	Logical position of Z-axis. It takes two registers. The MSW is located at address 94.
LP_U	96 (0x0060)	Logical position of U-axis. It takes two registers. The MSW is located at address 96.
EP_X	98 (0x0062)	Encoder feedback position of X-axis. It takes two registers. The MSW is located at address 98.
EP_Y	100 (0x0064)	Encoder feedback position of Y-axis. It takes two registers. The MSW is located at address 100.
EP_Z	102 (0x0066)	Encoder feedback position of Z-axis. It takes two registers. The MSW is located at address 102.
EP_U	104 (0x0068)	Encoder feedback position of U-axis. It takes two registers. The MSW is located at address 104.

Table 2: Holding register addresses for logical and encoder positions

7.1.2 Writing multiple register

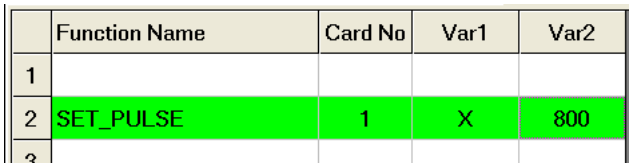
This format is being used for writing data to the remote device e.g. set velocity, acceleration, software limit etc. The majority of the motion functions sends data to the ET-M8194H device and therefore makes use of the Modbus function code 16.



No.	TxID	PID	Fld L.	UID	FC	St. Addr.	No.Reg.	BC	Reg. 1	Reg. 2	Reg. 3	Reg. 4
102	07 47	00 00	00 0F	01	10	1F 40	00 04	08	0A 4E	00 01	00 00	03 20
103	07 48	00 00	00 0B	01	10	1F 40	00 02	04	0A DB	00 01		
104	07 49	00 00	00 0F	01	10	1F 40	00 04	08	0A 4E	00 02	FF FF	FC E0
105	07 4A	00 00	00 0B	01	10	1F 40	00 02	04	0A DB	00 02		
106	07 4B	00 00	00 0F	01	10	1F 40	00 04	08	0A 4E	00 01	FF FF	FC E0
107	07 4C	00 00	00 0B	01	10	1F 40	00 02	04	0A DB	00 01		
108	07 4D	00 00	00 09	01	10	1F 40	00 01	02	0A CA			

No.	TxID	PID	Fld L.	UID	FC	St. Addr.	No.Reg.
102	07 47	00 00	00 06	01	10	1F 40	00 04
103	07 48	00 00	00 06	01	10	1F 40	00 02
104	07 49	00 00	00 06	01	10	1F 40	00 04
105	07 4A	00 00	00 06	01	10	1F 40	00 02
106	07 4B	00 00	00 06	01	10	1F 40	00 04
107	07 4C	00 00	00 06	01	10	1F 40	00 02
108	07 4D	00 00	00 06	01	10	1F 40	00 01

The following example (Table 3) shows the data field of the function call



	Function Name	Card No	Var1	Var2
1				
2	SET_PULSE	1	X	800
3				

SET_PULSE (BYTE cardNo, BYTE axis, DWORD data);

With

cardNo = 1
axis = 1 (x-axis)
data = 800

Request			Response		
Field Name	Length (Bytes)	(Hex) Example	Field Name	Length (Bytes)	(Hex) Example
Transaction ID:	2	07 47	Transaction ID:	2	07 47
Protocol ID:	2	00 00	Protocol ID:	2	00 00
Field Length:	2	00 0F	Field Length:	2	00 06
Unit ID	1	01 *(cardNo)*	Unit ID	1	01
Function Code:	1	10	Function Code:	1	10
Register **Starting Address**:	2	1F 40 *(8000)*	Register **Starting Address**:	2	1F 40 *(8000)*
Number of Register to be written:	2	00 04	Number of Register changed:	2	00 04
Number of bytes	1	08			
Value of Register (1F 40)	2	0A 4E *(SET_PULSE())*			
Value of Register (1F 41)	2	00 01 *(axis)*			
Value of Register (1F 42)	2	00 00 *(MSW data)*			
Value of Register (1F 42)	2	03 20 *(LSW data)*			

MSW- most significant word
LSW – least significant word

Table 3: SET_PULSE function call

8 Macro motion commands

The following table introduces the macros supported by the EzMove utility. The ET-M8194H\RS-M8194H manual describes these commands and their parameter setting in greater detail. The EzMake support three macro types:

- Macros for module initialization (IT)
- Macros for motion control (MP)
- Macros for interrupt service routine (ISR)

Some macro commands are only valid for a specific Macro program. The last three columns of the command table indicate whether the respective Macro program type supports the command.

8.1 Basic Setting Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	SET_PULSE_MODE	Sets the pulse output mode as either CW/CCW or PULSE/DIR for the assigned axes and their direction definition.	◎	◎		◎
	SET_MAX_V	Sets the maximum rate for the output pulses (speed). A larger value will cause a rougher resolution.	◎	◎		◎
	SET_HLMT	Sets the active logic level of the hardware limit switch inputs.	◎	◎		◎
	LIMITSTOP_MODE	Sets the motion stop mode of the axes when the corresponding limit switches are detected.	◎	◎		◎
	SET_NHOME	Sets the trigger level of the near home sensor (NHOME).	◎	◎		◎

	SET_HOME_EDGE	Sets the trigger level of the home sensor (HOME).	◎	◎		◎
	SET_SLMT	Sets the software limits.	◎	◎		◎
	CLEAR_SLMT	Clears the software limits.	◎	◎		◎
	SET_ENCODER	Sets the encoder input related parameters.	◎	◎		◎
	SERVO_ON	Outputs a DO signal (ENABLE) to enable the motor driver.	◎	◎		◎
	SERVO_OFF	Outputs a DO signal (ENABLE) to disable the motor driver.	◎	◎		◎
	SET_ALARM	Sets the ALARM input signal related parameters.	◎	◎		◎
	SET_INPOS	Sets the INPOS input signal related parameters.	◎	◎		◎
	SET_FILTER	Selects the axes and sets the time constant for digital filters of the input signals.	◎	◎		◎
	VRING_ENABLE	Enables the linear counter of the assigned axes as variable ring counters.	◎	◎		◎
	VRING_DISABLE	Disables the variable ring counter function.	◎	◎		◎
	AVTRI_ENABLE	Prevents a triangle form in linear acceleration (T-curve) fixed pulse driving even if the number of output pulses is low.	◎	◎		◎
	AVTRI_DISABLE	Disables the function that prevents a triangle form in linear acceleration fixed pulse driving.	◎	◎		◎

8.2 Status Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT

	SET_LP	Sets the command position counter value (logical position counter, LP).		◎	◎	
	GET_LP	Reads the command position counter value (logical position counter, LP).		◎	◎	
	SET_EP	Sets the encoder position counter value (real position counter, or EP).		◎	◎	
	GET_EP	Reads the encoder position counter value (EP).		◎	◎	
	GET_DI	Reads the digital input (DI) status.		◎	◎	
	GET_ERROR	Checks whether an error occurs or not.		◎	◎	
	GET_ERROR_CODE	Reads the ERROR status.		◎	◎	

8.3 FRnet DIO Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	FRNET_IN	Reads all the FRnet digital input signals of one group. One group comprises 16 bits data. Therefore, total 128 DI can be defined for one FRnet interface.		◎	◎	
	FRNET_OUT	Writes data to a FRnet digital output group. One group comprises 16 bits data. Therefore, total 128 DO can be defined for one FRnet interface.		◎	◎	
	FRNET_READ	Reads a single FRnet digital input and output signal of a selected group.		◎	◎	

	FRNET_WRITE	Writes a single FRnet digital output signal of a selected group.		◎	◎	
	FRNET_WAIT	Waits until the selected digital input changes it status to the wait signal		◎		

8.4 Auto Home Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	SET_HV	Sets the homing speed.	◎	◎		
	HOME _LIMIT	Sets the Limit Switch to be used as the HOME sensor.	◎	◎		
	SET_HOME_MODE	Sets the homing method and other related parameters.	◎	◎		
	HOME _START	Starts the home search of assigned axes.	◎	◎		

8.5 Axis Move Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	NORMAL_SPEED	The function sets the speed mode.	◎	◎	◎	
	SET_SV	Sets the start speed for the assigned axes.	◎	◎	◎	
	SET_V	Sets the desired speed for the assigned axes.	◎	◎	◎	

	SET_A	Sets the acceleration value for the assigned axes.	◎	◎	◎	
	SET_D	Sets the deceleration value for the assigned axes.	◎	◎	◎	
	SET_K	The function sets the acceleration rate (i.e., Jerk) value for the assigned axes.	◎	◎	◎	
	SET_L	Sets the deceleration rate (i.e., Jerk) value for the assigned axes.	◎	◎	◎	
	SET_AO	Sets the number of remaining offset pulses for the assigned axes. Please refer to the figure below for a definition of the remaining offset pulse value.	◎	◎	◎	
	FIXED_MOVE	Command a point-to-point motion for several independent axes.	◎	◎	◎	
	SET_PULSE	Sets the pulse number for fixed pulse driving.	◎	◎	◎	
	CONTINUE_MOVE	Issues a continuous motion command for several independent axes.	◎	◎	◎	

8.6 Interpolation Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	AXIS_ASSIGN	Assigns the axes to be used for interpolation. Either two or three axes can be assigned using this function. Interpolation commands will refer to the assigned axes to construct a working coordinate system. The X axis does not necessarily have to be the first axis. However, it is easier to use the X axis as the first axis, the	◎	◎	◎	

		Y axis as the second axis, and the Z axis as the third axis.				
	VECTOR_SPEED	Assigns the mode of interpolation. Either two or three axes will join this interpolation. Each interpolation mode will refer to some assigned axes that construct a working coordinate system. The assigned axes are defined by AXIS_ASSIGN() function. The X axis does not necessarily have to be the first axis. However, it is easier to let the X axis as the first axis, the Y axis as the second axis, and the Z axis as the third axis in applications. Different modes need different settings. Please refer to the mode definitions.	◎	◎	◎	
	SET_VSV	Sets the starting speed of the principle axis (axis 1) for the interpolation motion.	◎	◎	◎	
	SET_VV	Sets the vector speed of the interpolation motion. Users do not need to assign any axes on this function. The speed setting will take effect on the current working coordinate system which is defined by the AXIS_ASSIGN() function.	◎	◎	◎	
	SET_VA	Sets the vector acceleration for interpolation motion. Users do not have to assign any axes on this function. This speed setting will take effect on the current working coordinate system which is defined by the AXIS_ASSIGN() function.	◎	◎	◎	
	SET_VD	Sets the deceleration value for the interpolation motion.	◎	◎	◎	
	SET_VK	Set the acceleration rate (jerk) value for interpolation motion.	◎	◎	◎	
	SET_VL	Sets the deceleration rate of the interpolation motion.	◎	◎	◎	

	SET_VAO	Set this value will cause the motion control chip to start deceleration earlier. The remaining offset pulses will be completed at low speed to allow the controller to stop immediately when the offset pulse value has been reached. Please refer to the figure below for more information.	◎	◎	◎	
	LINE_2D	Executes a 2-axis linear interpolation motion.	◎	◎	◎	
	LINE_3D	Executes a 3-axis linear interpolation motion.	◎	◎	◎	
	ARC_CW	Executes a 2-axis circular interpolation motion in a clockwise (CW) direction.	◎	◎	◎	
	ARC_CCW	Executes a 2-axis circular interpolation motion in a counter-clockwise (CCW) direction.	◎	◎	◎	
	CIRCLE_CW	Executes a 2-axis circular interpolation motion in a clockwise (CW) direction.	◎	◎	◎	
	CIRCLE_CCW	Executes a 2-axis circular interpolation motion in a counter-clockwise (CCW) direction.	◎	◎	◎	

8.7 Synchronous Action Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	SYNC_ACTION	Sets the activation factors (provocative) and the specified action when a specified activation factor occurs.	◎	◎	◎	
	CLEAR_SYNC_ACTION	Clears all synchronous action of the selected axis	◎	◎	◎	

	SET_ACTIVATION_FACTORS	Sets the activation factors (provocative, trigger condition) of the monitoring axis.	◎	◎	◎	
	SET_ACTIVATION_AXIS	Set the axis which will execute an action when the monitoring axis activation factor has been met.	◎	◎	◎	
	SET_ACTION	Set the action of the activation axis	◎	◎	◎	
	SET_COMPARE	Sets the values of COMPARE registers. However, it will disable the functions of software limits.	◎	◎	◎	
	GET_LATCH	Gets the values from the LATCH register.	◎	◎	◎	
	SET_PRESET	Sets the PRESET value for synchronous action.	◎	◎	◎	
	SET_OUT	Sets the pulse by collocated the SYNC_ACTION function.	◎	◎	◎	

8.8 Continuous Interpolation Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	RECTANGLE	Continuous interpolation will be performed to create a rectangular motion, which is formed by 4 lines and 4 arcs. The length of each side can be changed. The radius of each arc is the same and it can also be changed. The deceleration point will be calculated automatically. This is a command macro command that appears in various motion applications.	◎	◎		

	LINE_2D_INITIAL	Sets the necessary parameters for a 2-axis continuous linear interpolation using symmetric T-curve speed profile.	◎	◎		
	LINE_2D_CONTINUE	Executes a 2-axis continuous linear interpolation.	◎	◎		
	LINE_3D_INITIAL	Sets the necessary parameters for a 3-axis continuous linear interpolation using symmetric T-curve speed profile.	◎	◎		
	LINE_3D_CONTINUE	Executes a 3-axis continuous linear interpolation.	◎	◎		
	MIX_2D_INITIAL	Does the initial settings for mixed linear and circular 2-axis motions in continuous interpolation.	◎	◎		
	MIX_2D_CONTINUE	Executes mixed linear and circular 2-axis motion in continuous interpolation.	◎	◎		
	HELIX_3D	Performs a 3-axis helical motion.	◎	◎		
	RATIO_INITIAL	Sets the Initial values for ratio motion (motion in ratio) using a symmetric T-curve speed profile.	◎	◎		
	RATIO_2D	Executes a two-axis ratio motion.	◎	◎		

8.9 Interrupt Control Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	ENABLE_INT	enables the interrupt.	◎	◎		
	DISABLE_INT	disables the interrupt.	◎	◎		

	INTFACTOR_ENABLE	sets the interrupt factors.	◎	◎	◎	
	INTFACTOR_DISABLE	disables the interrupt factors.	◎	◎	◎	

8.10 Other Functions

Icon	Function Name	Statement	RTC	MP	ISR	IT
	DRV_START	This command is usually used when users desire to start multi-axis driving simultaneously. When this command is issued, users may write other driving commands to the control card. All the driving commands will be held after DRV_HOLD() is issued, and these commands will be started once the DRV_START() is issued. However, if in driving, this command will not cause the driving to be stopped. But the next command will be held.	◎	◎	◎	
	DRV_HOLD	This command releases the holding status, and start the driving of the assigned axes immediately.	◎	◎	◎	
	STOP_SUDDENLY	Immediately stops the assigned axes.	◎	◎		
	STOP_SLOWLY	Decelerates and finally stops the assigned axes slowly.	◎	◎		
	CLEAR_STOP	Clears the stop status.	◎	◎		
	VSTOP_SUDDENLY	Stops interpolation motion of the assigned module immediately.	◎	◎		

	VSTOP_SLOWLY	Stops interpolation motion of the assigned module in a decelerating way.	◎	◎		
	CLEAR_**V**STOP	Clear the interpolation stop status.	◎	◎		
	EMERGENCY_STOP	Stops all axis.	◎			
	CLEAR_ EMERGENCY_STOP	Clear the emergency stop status.	◎			
	INTP_END	1. If the current motion status is running a interpolation motion and you would like to issue a single axis motion or change the coordinate definition, you should call this function before the new command is issued. 2.You can redefine the MAX V for each axis. In this way, you do not have to execute INTP_END() function.	◎	◎		

8.11 Macro Program Functions

Icon	Function Name	Statement	RTC	MP	ISR
	MP_CALL	Sets the number of the macro program for jump executing in the next procedure layer.		◎	
	MP_SET_VAR	sets VARn to be the global value.		◎	
	MP_SET_RVAR	sets VARn to be the global return value.		◎	
	MP_VAR_CALCULATE	issues the “addition”, “subtraction” “multiplication”, and the “division” for supporting the variable arithmetic operation. For		◎	

		example: varNo (+-*/) varNo1 = varNo2.			
	MP_FOR	issues the “for loop” condition statement.		◎	
	MP_NEXT	issues the “end of for loop” condition statement.		◎	
	MP_IF	issues the “If” condition statement.		◎	
	MP_ELSE	issues the “else” condition statement.		◎	
	MP_IF_END	issues the “end of If” condition statement.		◎	
	MP_TIMER	issues a procedure delay.		◎	
	MP_STOP_WAIT	can be used to assign commands to be performed while waiting for all motion to be completed (stopped).		◎	
	MP_SET_RINT	sets the i-8094H module to produce an interrupt signal with the 0x04 code to the WinCon controller.		◎	