Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

A GUIDE TO RS-232 COMMUNICATION WITH FX PLCS

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

A GUIDE TO RS-232 COMMUNICATION WITH FX PLCS

This document has been written specifically for FX and FX0N users that are unfamiliar with RS-232 'no-protocol' communication and would like to gain more understanding of what the function is and how it works. The aim of this document is not to be too technical, but to give explanations and examples[1] of key points where ever possible so that the reader can use the FX range in their own RS-232 system.

The intention is not to replace any of the associated manuals, such as the FX Programming Manual, or the FX-232ADP / FX0N-232ADP User Guides, but to back them up in learning how to use the function.

[1] Disclaimer:

All examples and diagrams in this document are included to aid the readers' understanding of the text. Mitsubishi Electric does not claim that these examples and diagrams are correct and error free, and will not accept responsibility for any consequential damage that may occur as a result of the installation and programming of the equipment associated with this document.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

Contents

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

1. INTRODUCTION TO SERIAL COMMUNICATION

This chapter introduces the concept of a serial communication system, and describes the RS-232 serial communication standard.

1.1 WHAT IS SERIAL COMMUNICATION?

Serial communication is the transfer of binary data (a *series* of 0's and 1's) through a single channel. Most devices like printers, computers, bar code readers, etc have a serial port by which they can be connected to another device with a serial port and the two devices can send and receive character or bit based data to each other.

A basic serial communication system is shown below, with one device sending data to the other.

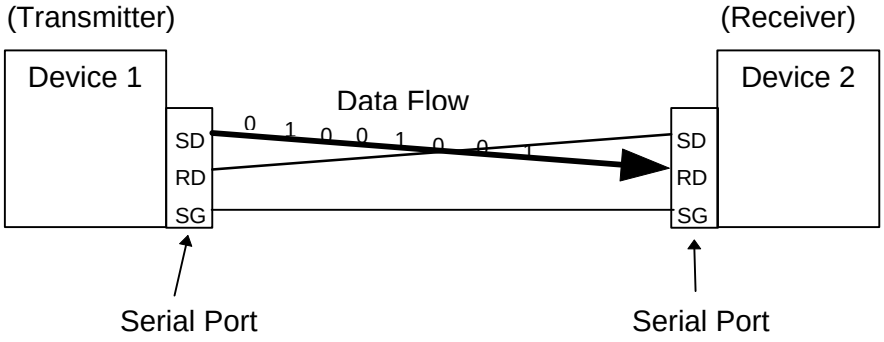


Figure 1-1 Basic serial communication system

Data is transmitted through SD (Send Data) of the transmitter, and received through RD (Receive Data) of the receiver. The voltage signals are relative to SG (Signal Ground).

1.2 TYPES OF SERIAL COMMUNICATION SYSTEM

Depending on the nature of the devices in the system, data will flow in a certain direction at a certain time. This section overviews the main types of serial communication.

1.2.1 Simplex

This describes a system where data flows in one direction only, such as a bar code reader connected to a terminal.

1.2.2 Half-duplex

This describes a system where data can flow in each direction, but only one way at a time. An example of a half-duplex system is a computer connected to a dot-matrix printer.

1.2.3 Full-duplex

This describes a system where data flows in each direction, both ways at the same time.
An example of a full-duplex system is two computers connected to each other.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

1.3 TYPES OF TRANSMISSION

There are two types of transmission; synchronous and asynchronous.

1.3.1 Synchronous Transmission

The transmitted data contains clock information that is used to synchronise the communication at both devices. This means that transmission overheads (i.e. start/stop bits) can be greatly reduced, as the transmission occurs at a specific time determined by the clock.

1.3.2 Asynchronous Transmission

The transmitted data does not contain clock information, and the receiver knows only the approximate speed of the transmission. Because the communication is not synchronised by a common clock signal, extra bits are required to signify the start and end of each character. These are called, unsurprisingly, the start bit and the stop bit.

This document will deal mainly with the asynchronous method, as it is the method used by the FX PLC.

1.4 WHAT IS RS-232?

RS-232 is the most common of the standards defining the physical and electrical interface of a serial port.

1.4.1 Physical interface definition

The physical interface defines the layout of the port and cable type. Below is the pin configuration for the 25-pin RS-232 port (there is also a 9-pin port). The maximum cable length is 15m.

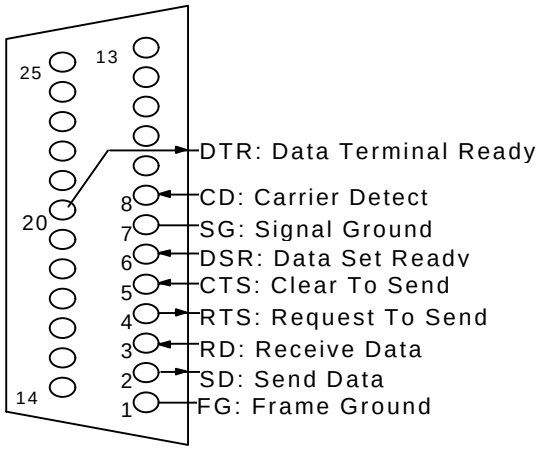


Figure 1-2 25-pin RS-232 port

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

The following table describes the operation of the signal pins of the RS-232 port.

Pin	Signal		Description
1	**FG**	Frame Ground	Used to provide grounding for the cable
2	**SD**	Send Data	Used to send data to the other device
3	**RD**	Receive Data	Used to receive data from the other device
4	**RTS**	Ready To Send	Indicates the device has data it wants to send
5	**CTS**	Clear To Send	Indicates the other device is ready to receive data
6	**DSR**	Data Set Ready	Indicates the other device is ready to receive data
7	**SG**	Signal Ground	All signals sent or received are relative to this pin
8	**CD**	Carrier Detect	Indicates that the carrier for transmission is on
20	**DTR**	Data Terminal Ready	Indicates the device has data it wants to send

1.4.2 Electrical interface definition

The electrical interface defines the voltage levels used to represent the 1s and 0s which make up the data.

For logic 0: +3 to +12V
For logic 1: -3 to -12V

(To allow for any noise in the channel, voltage signals between -3 to +3V are ignored.)

1.5 CONFIGURING COMMUNICATION

Both devices in a serial communications system need to be configured with the same communication parameters so that data flows effectively. This section will explain each parameter.

1.5.1 Data length

Transferred data is usually characters in an asynchronous transmission system. In most cases, these characters are in **ASCII** (American Standard Code for Information Interchange) format. The ASCII character set includes all the letters of the alphabet (upper and lower case), numbers 0 to 9, and other symbols such as +, =, % and so on (see Appendix 1 for the standard ASCII set).

Each ASCII character is encoded as a binary number so they can be tranmsitted through the serial channel. The number of bits that make up this binary number is referred to as the **data length**, and can be set as 7 bits or 8 bits.

When the standard ASCII set is used, the most significant bit in the number is redundant. So, by setting the data length as 7 bits, transmission overheads can be reduced (especially when lots of data needs to be transferred).

1.5.2 Baud rate

The baud rate is the speed of transmission, given in bits per second (bps). The baud rate can be set between 300 and 19200 bps.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

1.5.3 Start bit

The start bit, usually a 1, signifies the beginning of a character to the receiving device in an asynchronous system. This parameter is not usually configurable by the user.

1.5.4 Stop bit/s

Either one or two stop bits, usually a 0, is used to signify the end of a character in an asynchronous system. The user can choose whether one or two stop bits is added.

1.5.5 Parity bit

Parity is used as a simple method of error checking. The parity bit is used to make the number of 1s and 0s in the character either odd or even. This is illustrated below.

	Parity bit (even)	Parity bit (odd)
1 0 1 0 1 1 0	0	1
1 1 1 0 1 0 1	1	0

The idea is that if the parity is set, say to even for example, but the receiver counts an odd number of 1s, then it assumes that a bit is in error.

1.6 HANDSHAKING

Handshaking is a term which is used to describe how the flow of data is controlled by each of the devices in the system. There are two main methods; software and hardware.

1.6.1 Software handshaking

This takes place using certain control characters, such as XON / XOFF, which are sent between the devices to tell each other to begin or stop sending data.

A good example of this method is a laser-jet printer connected to a computer. When the printer buffer is empty, the printer sends XON to the computer. The computer will then start to send data to be printed. If the printer buffer becomes full, the printer sends XOFF, and the computer stops transmitting.

Other methods of software handshaking include the ENQ / ACK method, where the transmitter sends an ENQ (enquiry) command at the start of a **frame** (several characters grouped together), and the receiver sends back an ACK (acknowledge) to indicate the frame was received correctly. If a NAK (not acknowledge) is sent back instead, the transmitter resends the frame.

1.6.2 Hardware handshaking

This occurs using dedicated signal lines DSR & DTR and/or RTS & CTS within the serial port of the devices. DSR & DTR are perhaps more commonly used as the handshaking lines, as RTS & CTS are gradually being phased out in most systems.

When hardware handshaking is used, the DTRs and DSRs are connected together, as shown in the example RS-232 cable in Figure 1-3, known as a **null modem**.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

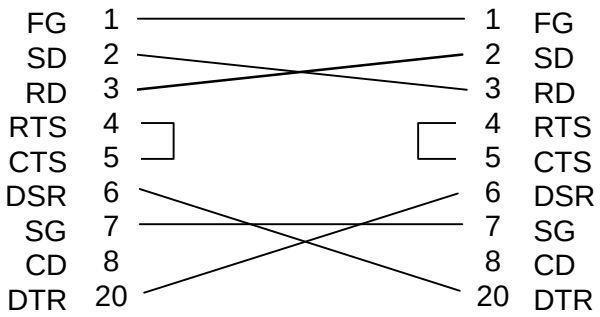


Figure 1-3 Example null modem cable

Normally, when a device has data it wants to transmit it turns on its DTR pin. The receiver then knows that the transmitter wants to send data, so when it is ready it also turns on its DTR pin. When the transmitter sees that its own DSR pin has come on, it will transmit. When the transmission has finished, the transmitter turns off its DTR to tell the receiver that no more data is pending. The receiver turns off its DTR, and the system is ready to begin again.

1.7 SUMMARY

A serial communication system transfers character or bit based data sequentially, one bit after the other, through a single channel.

The three types of serial system are simplex (one way only), half-duplex (two ways but one way at a time) and full-duplex (two ways simultaneously).

RS-232 is one of the standards for a serial port, and is perhaps the most common. The configuration is a 25-pin ‘D’ type port, but 9-pin ‘D’ type ports are also available (as many of the signal pins are unused in the 25-pin port). The RS-232 standard supports asynchronous communication (transmission is not synchronised by a clock signal, so extra bits are added to each character to signify their start and end).

Communication parameters, such as data length, baud rate, stop bits and parity are configurable by the user. Their settings are dependant on the setting requirements of the communication devices in the system.

Flow control is acheived using handshaking, either software or hardware orientated. With software handshaking, transfer of special characters controls the flow of data. With hardware handshaking, dedicated signal lines within the serial channel are used.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

2. OVERVIEW OF SERIAL COMMUNICATION WITH THE FX PLC

FX PLCs from version 3.30, and all FX0N versions, can be used as an asynchronous transmitter and/or receiver in an RS-232 serial communication system. The RS instruction defines message areas, and the FX-232ADP module provides the RS-232 port for the FX base unit; with the FX0N-232ADP for FX0N base units. To allow the user control of data transfer, some special auxilliary relays (M-coils) have been included. This chapter overviews how communication is acheived. More detail can be found in the next chapter.

2.1 FX-232ADP - RS-232 ADAPTER

Using the RS-232 adapter, the FX-232ADP, the FX PLC can be used to send and receive ASCII character or bit data in a serial communication system. The FX-232ADP can be seen below.

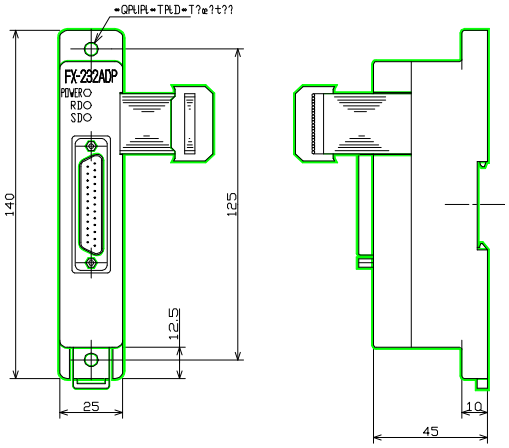


Figure 2-1 The FX-232ADP RS-232 adapter module for FX PLCs

The module plugs into the communication port on the left hand side of the FX base unit. The pin out is the same as the ‘standard’ pin out that is shown in Figure 1-2 on page 5. The FX0N-232ADP is the FX0N equivalent to this module.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

2.2 USER DEFINED 'NO-PROTOCOL' COMMUNICATION

What makes using serial communication with the FX range so flexible is that it does not use a specific protocol* or port setting. The user has the freedom to create message frames within the sequence program, and set the communication parameters in a special data register. Hence the FX can be made to match the protocol and port setting of virtually any other RS-232 device.

The next chapter details how the user can set up and use the serial communication function.

* *Protocols define the communication structure necessary for succesful data transfer, i.e. frame format (character sequence) and handshaking method.*

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

3. CREATING THE SERIAL COMMUNICATION PROGRAM

The basic format of the serial communication sequence program is shown in Figure 3-1 below. Explanations of each main stage follow.

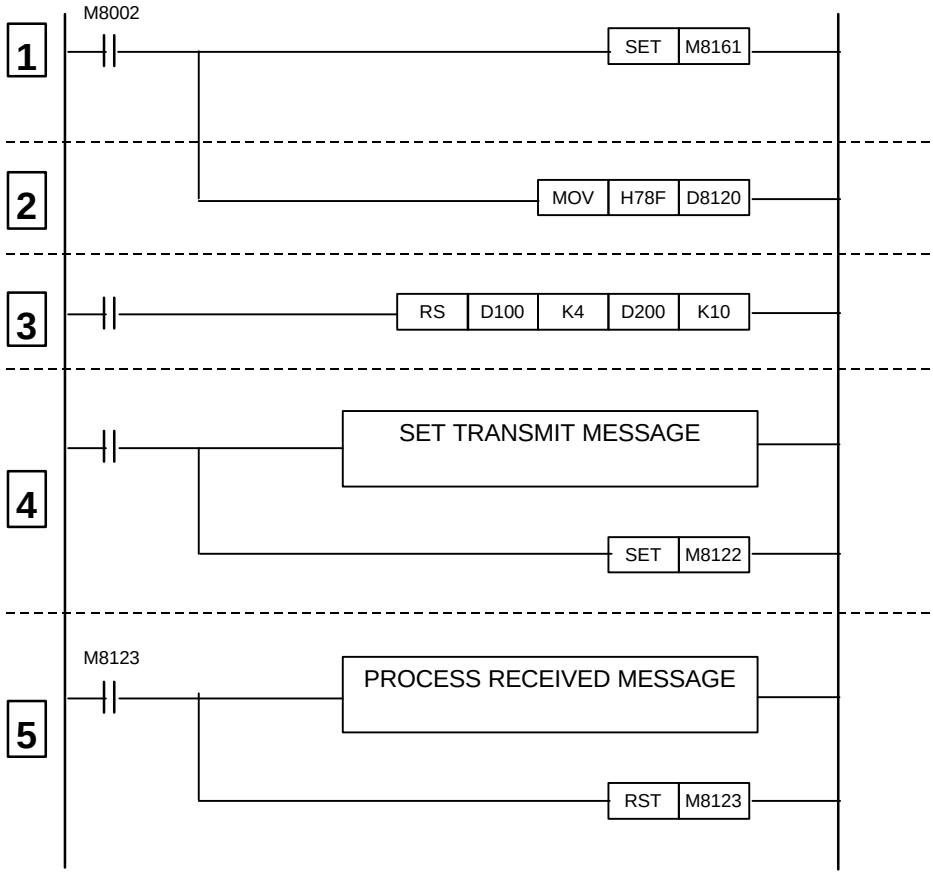


Figure 3-1 Basic format of sequence program

3.1 DATA STORAGE FORMAT

The data storage format refers to whether one or two ASCII characters is held in a single 16-bit data register.

The flag M8161 is used to set the format; ON for 8-bit storage (one character), OFF for 16-bit storage (two characters). When 8-bit storage is selected, only the lower 8 bits of 16-bit data registers are used to hold data. The upper 8 bits are not used. If 16-bit is selected, then all 16 bits of data registers are used to hold data.

Figure 3-2 illustrates 8-bit and 16-bit data storage of ASCII characters A B C D, in registers starting at D10 :

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

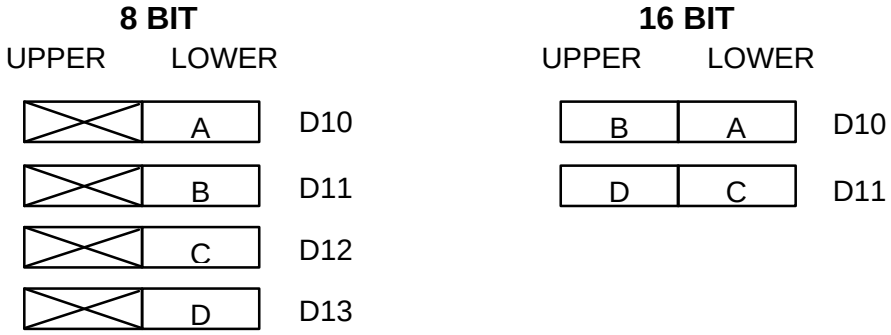


Figure 3-2 Data storage format example

Although 8-bit storage makes the message look easier to read, 16-bit storage uses half the number of data registers - particularly useful if several large messages are to be stored.

3.2 SET COMMUNICATION PARAMETERS

As mentioned previously, it is necessary to configure both RS-232 devices to the same communication parameters, such as number of stop bits, data length, baud rate, etc, or there will be communication errors. This information is written into data register D8120, and the value, simplest in hexadecimal, to write into this data register is derived by first constructing a 'bit map' (the allocation of 1's and 0's of the bits that make up the data register) of D8120, based on the information in the table below.

	D8120		
bit	**Description**	**0 (off)**	**1 (on)**
b0	Data Length	7 bits	8 bits
b1 b2	Parity (bits b1 and b2)	(00) (01) (11)	No Parity Odd Parity Even Parity
b3	Stop bits	1 bit	2 bits
b4 b5 b6 b7	Baud Rate (bps) (bits b4, b5, b6, b7)	(0011) (0100) (0101) (0110) (0111) (1000) (1001)	300 bps 600 bps 1200 bps 2400 bps 4800 bps 9600 bps 19200 bps
b8	Header Character	none	Default STX (in D8124)
b9	Terminator Character	none	Default ETX (in D8125)
b10	Handshake type 1	none	used (hardware)
b11	Handshake Mode	Normal mode	Single line mode
b12	Handshake type 2	none	used (hardware)
b13	NOT USED		
b14	NOT USED		
b15	NOT USED		

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

3.2.1 Parameter set up example

To set into the FX the following communication parameters:

Data Length:	8 bits
Parity:	Even
Baud rate:	9,600 bps
Header:	Used
Terminator:	Used
Handshake:	type 1
Handshake Mode:	Ordinary

D8120

b15 b0

The bit map would be: 0 0 0 0 0 1 1 1 1 0 0 0 1 1 1 1 = 078F (hex)

As the ladder diagram in Figure 3-1 on page 12shows, this can be written into D8120 using the MOV instruction.

3.2.2 Header and terminator characters

Bits b8 and b9 can be used to attach predefined characters automatically to the start and end of a message. These characters are the *header* and *terminator* respectively. The header is written to D8124, and the terminator to D8125. Both are user definable, and the defaults are STX (start of text) and ETX (end of text).

3.2.3 Hardware handshaking types

There are two types of hardware handshaking available, both using DSR and DTR. Only one may be selected at a time. The difference between the two types is the timing of how long the DSR and DTR lines are high (on) and low (off). Refer to the timing diagram in Appendix 3.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

3.3 THE RS INSTRUCTION

The RS instruction has 5 parts to it, as shown in Figure 3-3.

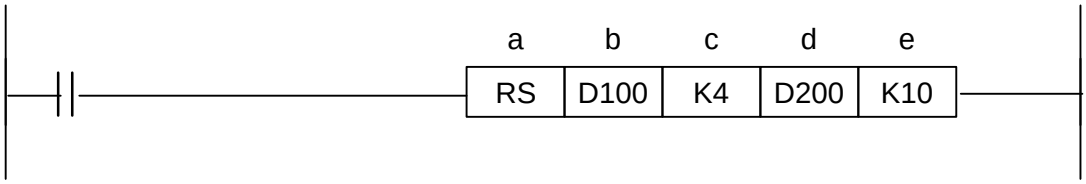


Figure 3-3 The RS instruction layout

a) The RS statement - This must be active for data transfer to occur.

b) The transmit message buffer head address - This is the data register that stores the first character of a message to be transmitted by the FX.

c) The transmit message buffer length - This defines the number of characters in the message that is to be transmitted. Each character will be stored sequentially in a stack of data registers. The value of the buffer length can be a decimal 'K' value, as shown in Figure 3-3, hexadecimal 'H' value, or a data register 'D' value. It is useful to use data registers if the message length will vary.

d) The receive message buffer head address - This is the data register where the first character in a message that is being received by the FX will be held.

e) The receive buffer length - this defines how many characters will be in the message that the FX will receive. Each received character will be stacked sequentially in data registers until the receive buffer becomes full. As with transmit buffer length, the value can be a 'K', 'H' or 'D'.

3.4 SET TRANSMIT MESSAGE

This basically means write or copy, using MOV or BMOV statements, the transmission message into the data registers you have assigned as the transmit buffer.

When the data is in the buffer, it is necessary to activate the TRANSMIT MESSAGE flag M8122 to actually send the data through the FX-232ADP to the peripheral RS-232 device. The flag is automatically reset when the message has been transmitted.

3.5 PROCESS RECEIVED MESSAGE

When a complete message has been received by the FX (the receive buffer has become full, or the character defined as the terminator has been received) the MESSAGE RECEIVED flag M8123 comes ON. This flag can then be used to drive a BMOV statement that will copy the received message to another location, where it can be processed. The flag will need to be reset so that other messages can be received.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

4. POINTS TO NOTE ABOUT FX SERIAL COMMUNICATION

- The FX-232ADP can only be used to transfer data through the communication port (on the left hand side of the FX base unit), and is used in conjunction with the RS instruction. It cannot be used to access, transfer or edit ladder program.

- The programming port is left free when using the FX-232ADP, allowing programming tools, MMI's, etc to still be connected.

- Modem connection is made easier by automatic use of the CD (Carrier Detect) line. This basically means that when the FX is dialling out through a modem, it will be able to detect when the telephone-line connection has been made (M8124 comes ON). This is feature with the FX only, not FX0N.

- If more than one message is to be transfered, separate RS instructions are needed for each message. Only one RS instruction may be active at a time.

- To supplement the RS instruction, there are instructions to calculate sum check code[**] (CCD instruction)**,** convert ASCII characters to hexadecimal digits (HEX instruction), and convert hexadecimal digits to ASCII characters (ASCI instruction).

  More information on these instructions can be found in the FX Programming Manual.

- If headers and/or terminators are selected, these are automatically added to a message by the FX.

- When using 16-bit operation, the data in the lower half of each data register is sent first.

- 16 bit operation means that half as many data registers are used for the transmit and/or receive buffers.

[**] Sum checking is a method of error detection. A sum check code is calculated by *summing* the hexadecimal values of each character in the message frame, within the sum check range. The transmitting device calculates the sum check code, and adds it to the end of the data. The receiving device will perform a sum check of the received data, and if its sum check code is different, it will signify an error.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

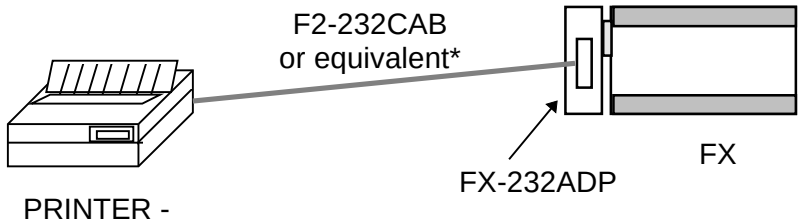
5. APPLICATION EXAMPLES

This chapter includes some suggested applications examples for FX serial communication. All the hardware, and software information is given and it is intended that the examples are used **as a guide** to producing your own serial communication system.

5.1 SENDING MESSAGES TO A PRINTER WITH THE FX

The FX can be connected to a printer, to print a brief report or a message whenever it is necessary, such as error reports or production quotas, etc.

The example set up is as shown below:

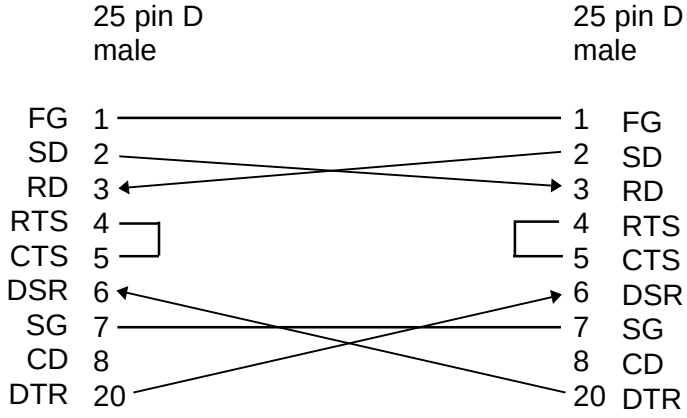


Baud - 9600
Data bits - 8
Stop bits - 1
Parity - None

D8120 in the PLC needs to set as:

bit															bit
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
	0				0				8				1		hex

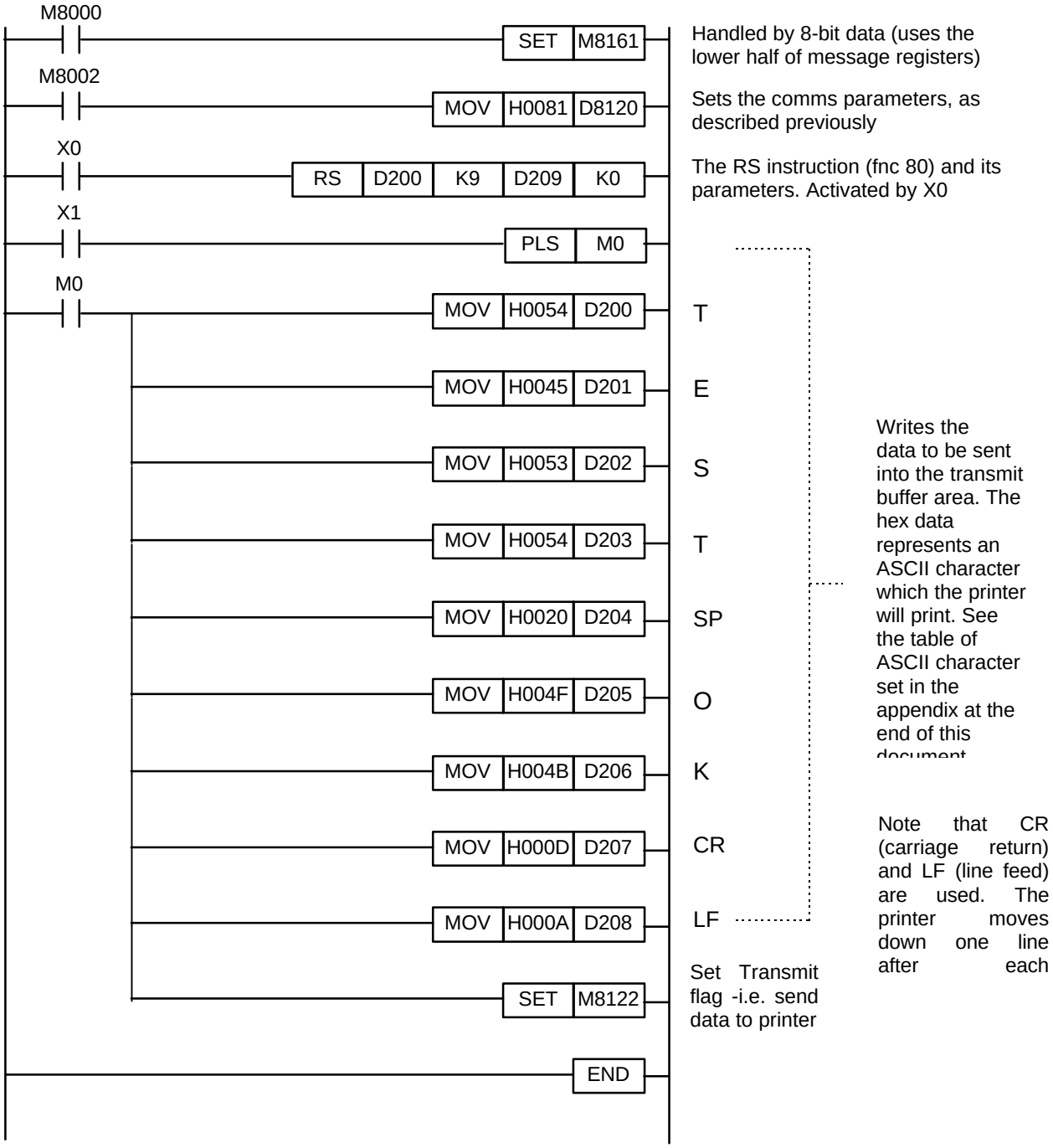
*The pin configuration for F2-232CAB is:



To print a simple message

In the following example, the message “TEST OK” will be printed whenever X1 is activated. The ladder program into the PLC would therefore be:

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew



How it works

When the PLC and printer are powered up and connected as shown in the diagram on the previous page, with the printer on-line and the PLC set to RUN, the message will be sent if the RS instruction is activated and the transmit flag is ON. In the example, X0 drives the RS instruction, and X1 pulses M0, which writes the message “TEST OK” into the message area, as defined in the RS instruction parameters (D200 - D208). M0 also sets M8122, thus sending the message. M8122 is automatically reset when the data has been sent, so the message will be sent to the printer every time X1 is triggered.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

Creating your own messages

Any message is a group of ASCII characters. The ASCII character set includes all the letters of the alphabet, upper and lower case, and all numbers 0-9 and most symbols. In the example given, the message 'TEST OK' was created by looking at the table given in the appendix and finding the hexadecimal value for each character in the message.
All messages can be created in this way. Be sure to note down the quantity of characters in the message, so it can be used as the transmit buffer length in the RS instruction.

Things to note when sending a message to a printer

- The communication parameters in the PLC must match those of the printer.

- Each ASCII character is represented as 2 hexadecimal digits. When data storage is set as 16 bit (M8161 OFF), then each data register in the transmit buffer can contain 2 ASCII characters as hexadecimal codes (one character in the upper half, one in the lower. The lower half of each register is sent first). When in 8 bit mode (M8161 ON) each data register in the transmit buffer can contain only 1 ASCII character (in the lower half. The upper half is ignored).

- The receive buffer head address must be assigned in the RS instruction, even though the PLC will not receive any data from the printer. Set the receive message length to 0 bytes.

- If more than one message is to be sent, separate RS instructions are needed for each message. Only one RS instruction may be active at a time.

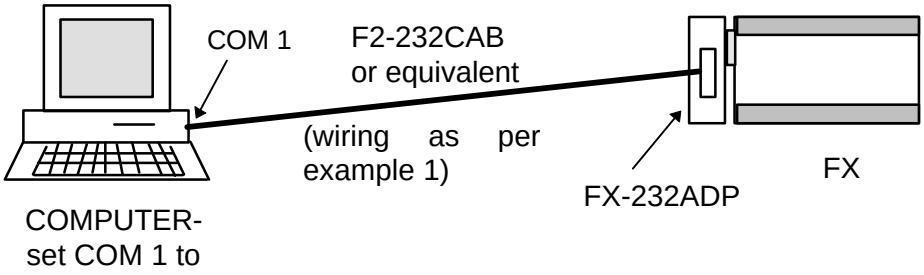
- RAM file registers (D6000 - D7999) can be used to store ASCII text strings. If it is Medoc (ver 1.63 or greater) that is being used to program the FX, ASCII characters can be directly written into RAM file registers under the 'DWRset' editor screen. When in the working area, press function key F10 to select ASCII mode. Then, the desired characters can simply be typed directly into the registers. The BMOV instruction can be used to move the particular message to be sent into the transmit buffer (remember to turn on the RAM file registers activate flag M8074). Perhaps X0 loads one message, X1 loads another, etc. (Don't forget to download the DWRset to the FX.)

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

5.2 SENDING AND RECEIVING DATA TO/FROM A PC

The FX can be made to receive a text message directly from a PC. This example uses the software Terminal Emulator on the PC.

The example set up is shown below:



Baud - 9600
Data bits - 8
Stop bits - 1
Parity - None
Flow control* - None

D8120 in the PLC needs to set as:

bit															bit
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
	0				0				8				1		hex

*Flow control = handshaking

Setting up Terminal Emulator

To run Terminal Emulator:

In ‘Program Manager’, open the group ‘Accesories’. Double-click the ‘Terminal’ icon.

Or

In ‘Program Manager’, from the File menu select Run. Type ‘terminal.exe’

To set up the communication parameters:

From the Settings menu select Communications. Select the parameters as described in the diagram above. Also, from the Settings menu select Terminal Emulation. Select DEC VT-100 [ANSI].

Sending a message to the computer from the FX

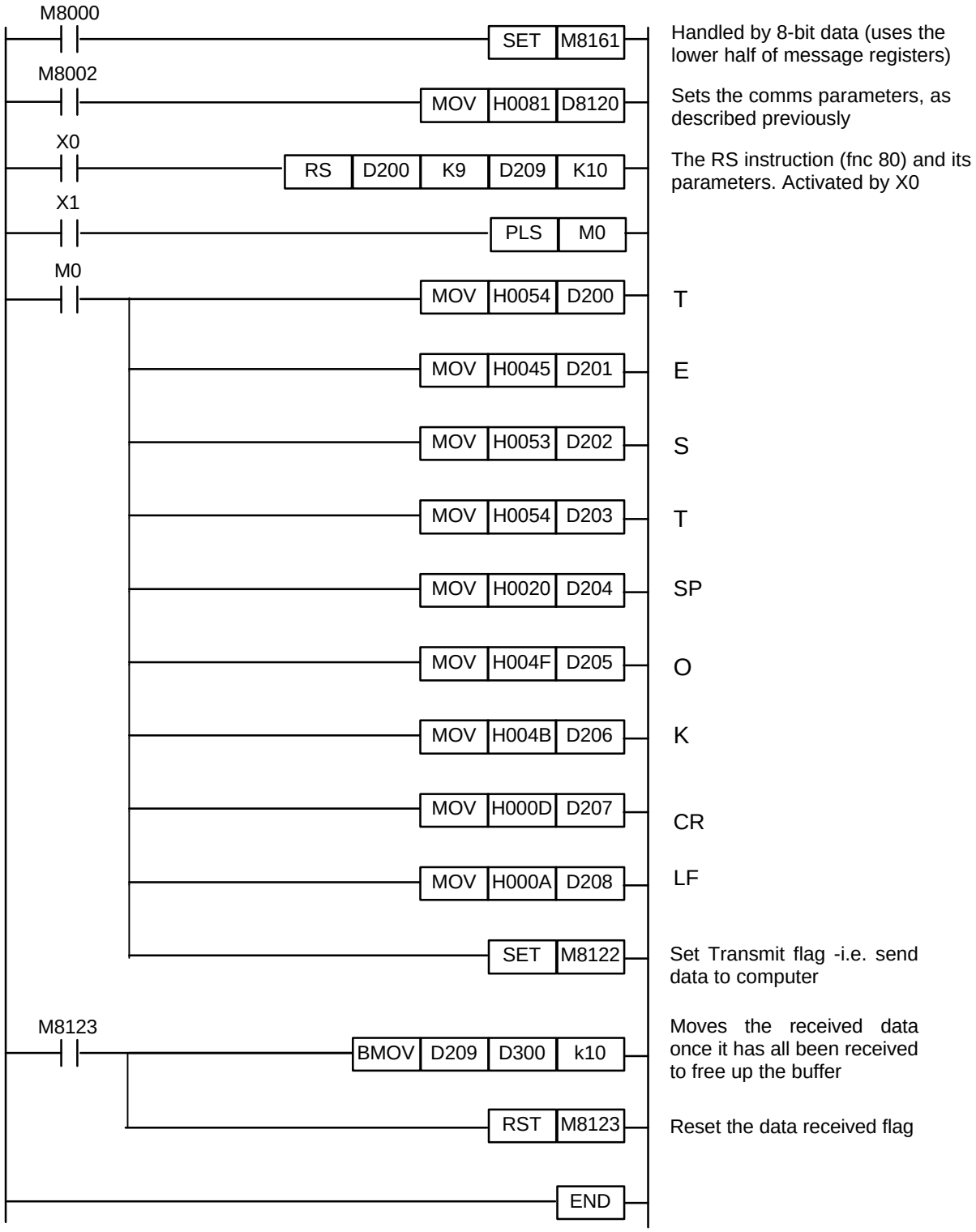
The program from the previous example can be used here to make ‘TEST OK’ appear on the computer screen each time X1 is pulsed.

Sending characters to the FX from the computer

With Terminal Emulator, each time a key is pressed, the character that corresponds to that key will be sent out of COM 1 to the FX via the FX-232ADP. So, to send the message ‘NEW FX PLC’ to the PLC use the following ladder program and type in capitals ‘NEW FX PLC’.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

The following example ladder program performs the operation of sending 'TEST OK' to the computer as well as being able to receive messages from the computer.



Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

How it works

The sending of 'TEST OK' is the same as in the previous example. The message will appear on the computer screen each time the input X1 is pulsed.

For the receiving of data, when all of the data has been received (i.e. when the 10 characters have been typed) then the flag M8123 will come on and move 'NEW FX PLC' into data registers D300 to D309 to free the buffer for the next lot of data. The flag M8123 is automatically reset by the FX ready for the next lot of data.

Things to note

- Although the communication parameters are more flexible for this application, the parameters in the PLC and the computer must match.

- If possible monitor, using the device monitor function in Medoc, what is coming into the receive buffer. When the expected number of characters has been received, it will be possible to see the message get copied to their new location.

- The example uses the message 'NEW FX PLC' arbitrarily. Any key can be pressed, and it's corresponding ASCII character will be sent to the FX. Just remember that the receive buffer will be full when the expected number of characters has been received (the receive buffer length), or the character designated as the terminator has been received (only when terminator is selected).

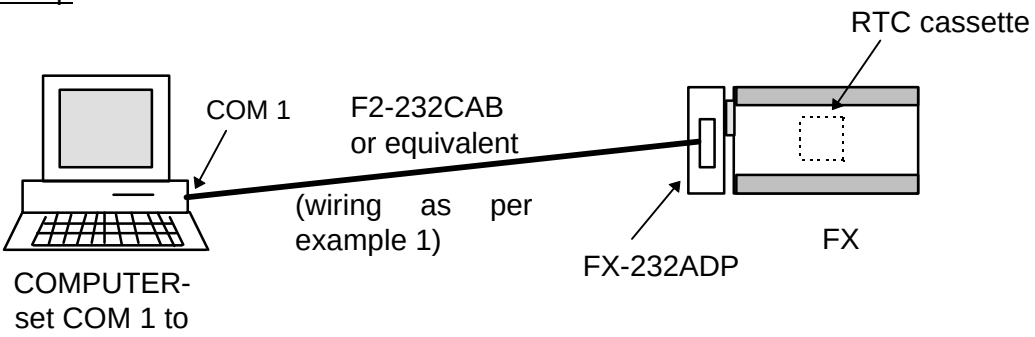
- Messages from the FX can be saved on the computer by first selecting 'Receive text file' from the Transfers menu on Terminal Emulator. It will ask what to call the file as a text file ([filename].txt) and where to save it. Terminal Emulator will stop receiving the text file when the 'STOP' button has been pressed on the screen. This could be useful if you wanted to keep a log of messages from the FX.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

5.3 PRINTING REAL TIME CLOCK (RTC) DATA - USING THE ASCI INSTRUCTION (FNC 82)

Time and date information may need to be included as part of a message. This data can be obtained from one of the RTC cassettes (FX-RTC, FX-EEPROM-4C, FX-EEPROM-8C). The following example shows a method of converting the time data (hours, minutes and seconds) from the RTC into printable ASCII characters.

System set-up



Baud - 9600
Data bits - 7
Stop bits - 1
Parity - None
Flow control - None

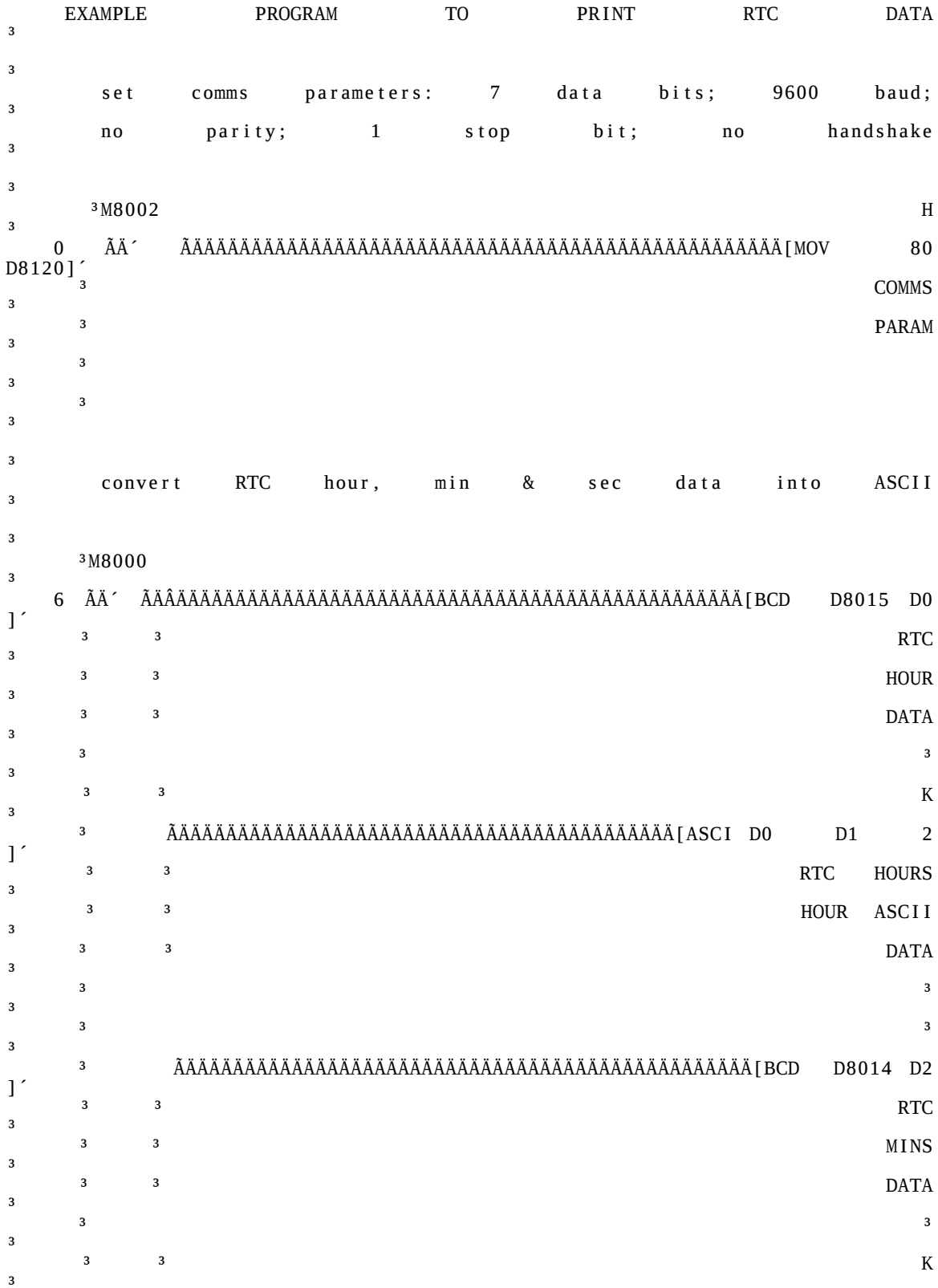
D8120 in the PLC needs to set as:

bit															bit
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
	0				0				8				0		hex

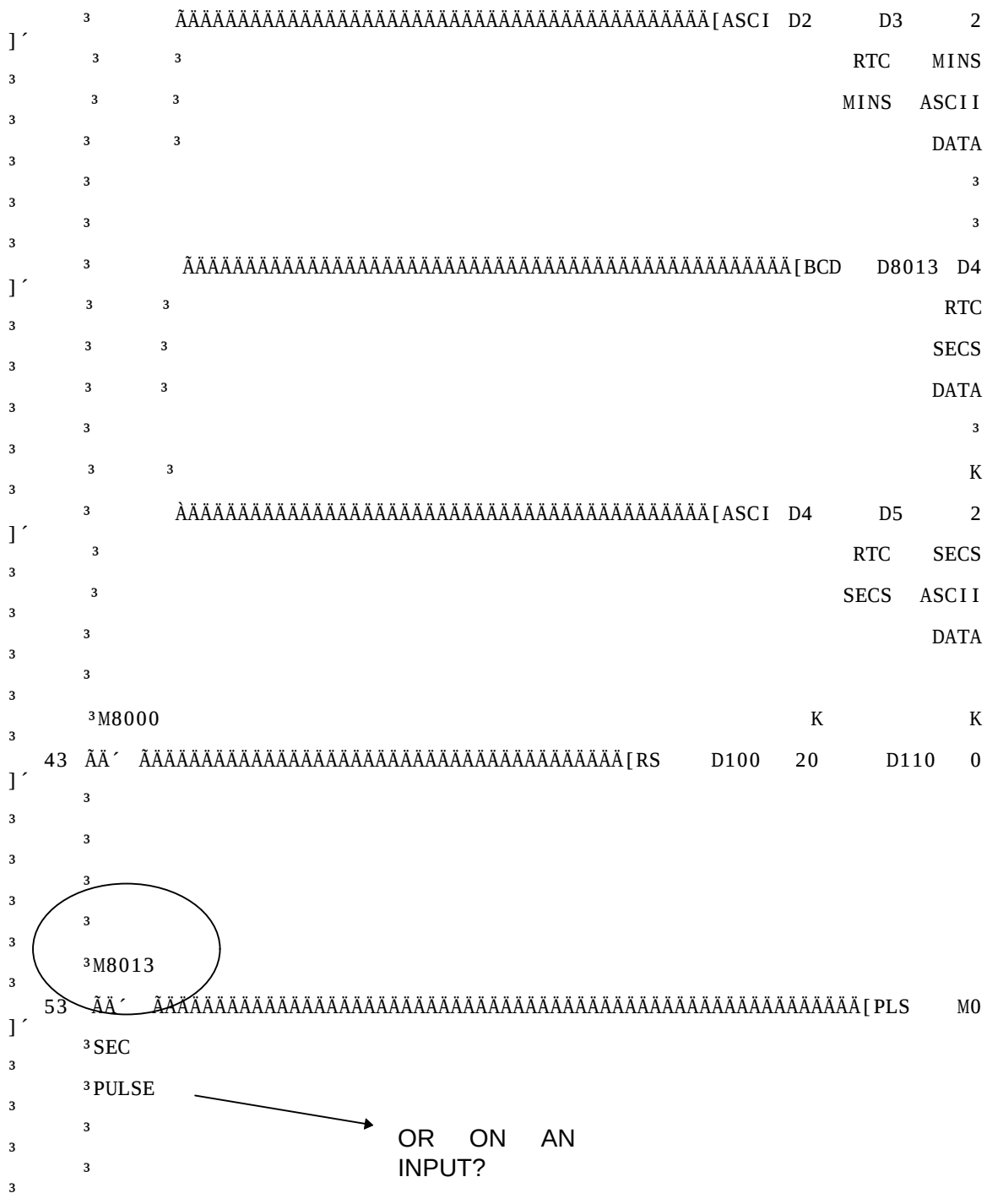
Terminal Emulator is used as per example 5.2. Note the change in communication parameters (7 data bits this time).

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

Ladder program



Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew



message "TIME IS __: __. __[CR][LF]" is sent every second

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

```
          ³M0                                                                 H
³
   56  ÃÄ´  ÃÄÂÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   4954   D100
]´
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
          ³     ³                                                             H
³
         ³     ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   454D   D101
]´
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
          ³     ³                                                             H
³
         ³     ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   4920   D102
]´
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
          ³     ³                                                             H
³
         ³     ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   2053   D103
]´
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
         ³     ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   D1     D104
]´
         ³     ³                                                          HOURS
³
         ³     ³                                                          ASCII
³
        ³                                                                     ³
³
        ³                                                                     ³
³
          ³     ³                                                             H
³
         ³     ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   203A   D105
]´
        ³                                                                     ³
³
        ³                                                                     ³
³
        ³                                                                     ³
³
```

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

```
        3                                                                                3
3
        3                                                                                3
3
        3       ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   D3      D106
]´
        3       3                                                                MINS
3
        3       3                                                               ASCII
3
        3                                                                                3
3
        3                                                                                3
3
        3       3                                                                        H
3
        3       ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   202E    D107
]´
        3                                                                                3
3
        3                                                                                3
3
        3                                                                                3
3
        3                                                                                3
3
        3                                                                                3
3
        3       ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   D5      D108
]´
        3       3                                                                SECS
3
        3       3                                                               ASCII
3
        3                                                                                3
3
        3                                                                                3
3
        3       3                                                                        H
3
        3       ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV   A0D     D109
]´
        3                                                                                3
3
        3                                                                                3
3
        3                                                                                3
3
        3                                                                                3
3
        3                                                                                3
3
        3               ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[SET
M8122]´
        3
3
        3
3
        3
3
        3
3
        3
3
  109   ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[END
]´
```

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

How it works

When a RTC cassette is connected, the time data is automatically written to specific data registers within the FX CPU. The program example takes only the time data, which is automatically written to D8015 (hours), D8014 (mins) and D8013 (secs).
The data in each of these registers is continually converted to BCD form, and then converted to ASCII using the ASCI instruction. Using the second pulse flag, M8013, the message "TIME IS 'HH': 'MM'. 'SS'" is displayed on the computer screen every second.

Things to note

- The main aim of this example is to show how to insert RTC data, which is not fixed data, into a fixed message. It is useful mostly, perhaps, when the FX is connected to a computer or printer to display/print a production log, or error report, so that it contains the date and time of the error, etc.

- It is necessary to firstly convert each part of the time information to BCD format because the ASCI instruction converts **hexadecimal digits** to ASCII characters. Conveniently, each **digit** in the time information will be in the range 0-9 (there will never be, for example, a time 03: 2B. 1A). So BCD form, for this example, will be like hexadecimal without the digits A-F. This data can be 'put through' the ASCI instruction quite easily.

- The ASCI instruction puts the ASCII characters in the correct order for sending. As this example operates in 16-bit data mode, it will be the lower byte of each register in the tranmsit buffer that is sent first. So to send, for example, the data for 19 mins, we obviously want it to appear at the receiving device (a computer in this case) as "19" mins, and not "91" mins. The ASCI instruction will convert BCD 19 and store it as ASCII 9 (upper byte), ASCII 1 (lower byte). So the tenths data is sent first, followed by the units.

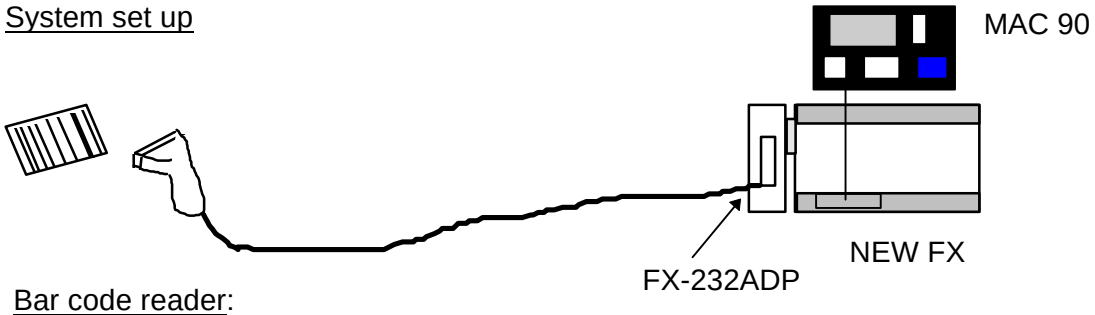
- The example uses the special flag M8013 (the one second pulse flag), so that the message is sent to the computer every second, so that it appears that the computer is being used as a clock display. Instead of M8013, an input or an M coil can be used to send the data when it is required (especially when a printer is used).

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

5.4 READING BAR CODES

Many applications require the need to scan a bar code - production monitoring, inventory control, etc. A bar code reader is an RS-232 device that scans a bar code and converts it to an ASCII string. The following example shows how to a bar code can be scanned, and then displayed on a MAC 90 terminal.

System set up



Bar code reader:

data bits:	8
parity:	none
stop bits:	1
baud:	9600
header:	used
terminator:	used
handshake:	none

Ladder program

```
     Set                  up                  comms                  parameters
3
3
       ³M8002                                                                    H
3
    0   ÃÄ´   ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[MOV      381
D8120]´
       3
3
       3
3
       3
3
       3
3
       3
3
     Set               up              8              bit               mode
3
3
       ³M8000                                                                 M8161
3
    6   ÃÄ´   ÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ(
)ÄÄ´
       3
3
       3
3
       3
3
       3
3
```

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

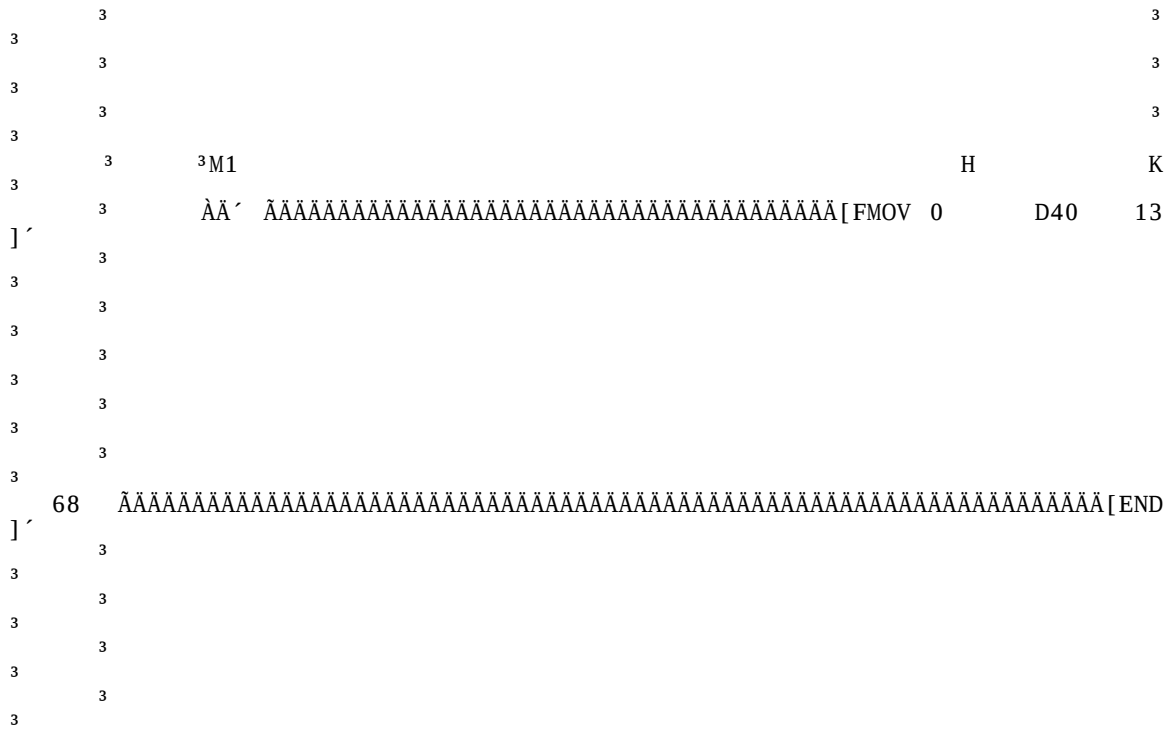
```
³     Actual send and receive instruction
³
³      ³M8000                                        K       K
³   9  ÃÄ´  ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[RS   D100  1   D0   20   ]´
³        ³
³        ³
³        ³
³     move data from d0 to d20
³
³      ³M8123                                                          K
³  19  ÃÄ´  ÃÄÂÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[BMOV  D0   D20   20   ]´
³        ³                                                              ³
³        ³                                                              ³
³        ³                                                              ³
³        ³                                                              ³
³        ³                                                              ³
³        ³          ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[RST M8123]´
³        ³
³        ³
³        ³
³        ³
³     reset V register
³
³      ³M8002
³  29  ÃÄ´  ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[RST    V   ]´
³        ³
³        ³
³        ³
³        ³
³     For next loop
³
³       ³                                                              K
³  33   ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[FOR    13   ]´
³        ³
³        ³
```

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

```
        ³
³
        ³
³
    Strip              off              upper              12              bits
³

³
        ³M8000                                                   H         V      V
³
   36  ÃÄ´  ÃÄÂÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[WAND  F        D20     D40
]´
        ³                                                                          ³
³
        ³                                                                          ³
³
        ³                                                                          ³
³
        ³                                                                          ³
³
        ³                                                                          ³
³
        ³        ÀÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[INC      V
]´
        ³
³
        ³
³
        ³
³
        ³
³
        ³
³
   47
ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[NEXT]´
        ³
³
        ³
³
        ³
³
        ³
³
    Reset                                  V                                 register
³

³
        ³M8000
³
   48   ÃÄ´   ÃÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[RST       V
]´
        ³
³
        ³
³
        ³
³
        ³
³
    if     reading     invalid     write     0000     to     all     13     words
³

³
        ³M8000                                                                     H
³
   52  ÃÄ´  ÃÄÂÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄÄ[CMP    3F       D20     M0
]´
        ³                                                                          ³
³
```

MITSUBISHI ELECTRIC

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew



How it works

When a bar code is scanned, the individual characters are stored in data registers D40 to D52. Each data register can be monitored using the MAC 90, and the 13 digit bar code can be displayed on the screen.

To display a data register value on the MAC 90, set up an ANALOGUE NUMERIC object. The complete code can be displayed when each of the 13 data registers (D40-D52) are displayed as analogue numeric objects next to each other in the same block.

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

Appendix 1 - Standard ASCII character set

HEX	COLUMNS	STANDARD TABLE FOR HEX TO ASCII CONVERSION						
ROWS	0	1	2	3	4	5	6	7
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

The table above shows how to find the hex code for ASCII characters. The two digit hexadecimal code for each character is expressed [column, row]. E.g. the code for the ASCII character ‘A’ is 41H.

Note: The cells in the table with a heavy border are the ASCII character set most relevant for use with ‘no-protocol’ communication.

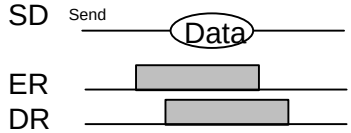
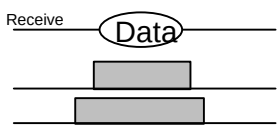
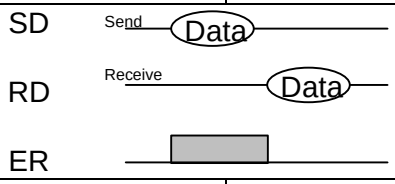
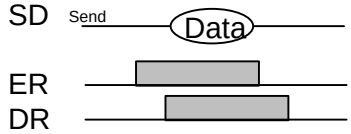
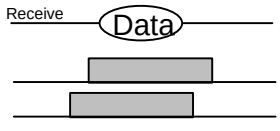
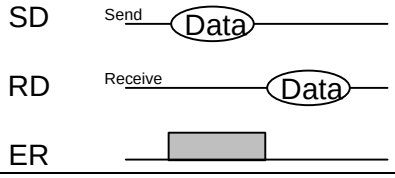
Appendix 2 - converting 25 pin to 9 pin connectors

The table below shows the equivalent pins for a 9-pin D-sub connector to that of the 25-pin D-sub connector, for RS-232 communication. This can be used as a guide for creating cables.

Signal	25 pin	9 pin
FG	1	
SD	2	3
RD	3	2
RTS	4	7
CTS	5	8
DSR	6	6
SG	7	5
CD	8	1
DTR	20	4

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

Appendix 3 - Timing Chart For Handshaking Methods

Handshaking method Selected in D8120	Send	Receive
b10: 1 b11: 0 Hardware 1, Normal mode		
b10: 1 b11: 1 Hardware 1, Single line mode		
b12: 1 b11: 0 Hardware 2, Normal mode		
b12: 1 b11: 1 Hardware 2, Single line mode		

ER= DTR DR= DSR

Appendix 4 - Further Reading

JY992D69901
FX Communications Manual (RS232, RS485)
User's Manual

JY992D66701
FX2N-232IF RS-232C Interface Block
User Manual

JY992D66001
FX2N-232-BD Communications Board
User Guide

Issued by: Mitsubishi Electric Europe B.V. - UK Branch
Document issue: Ver B, August 1998
Produced by : - Customer Technology Centre (UK) (CTC UK)
Travellers Lane, Hatfield, Herts, AL10 8XB, UK
Author: Alex Kew

NOTES