DUEMMEGI

SISTEMI DI AUTOMAZIONE INDUSTRIALE

CONTATTO

ISD – Data Exchange Interface

User's Manual

DUEMMEGI s.r.l. - Via Longhena, 4 - 20139 MILANO
Tel. 02/57300377 - FAX 02/55213686

Rel.: 1.0
June 2002

INDEX

1- INTRODUCTION

ISD is a module of CONTATTO bus family. ISD means "**I**nterfaccia per **S**cambio **D**ati" (Data Exchange Interface) and it allows the handling and the exchange of information between more MCPs (up to **31**), each one having its field bus, connected together and to ISD in a RS485 network. To allow the data exchange between a MCP and another one, **ISD must be programmed** through simple equation in a similar way to those used by MCP controllers.

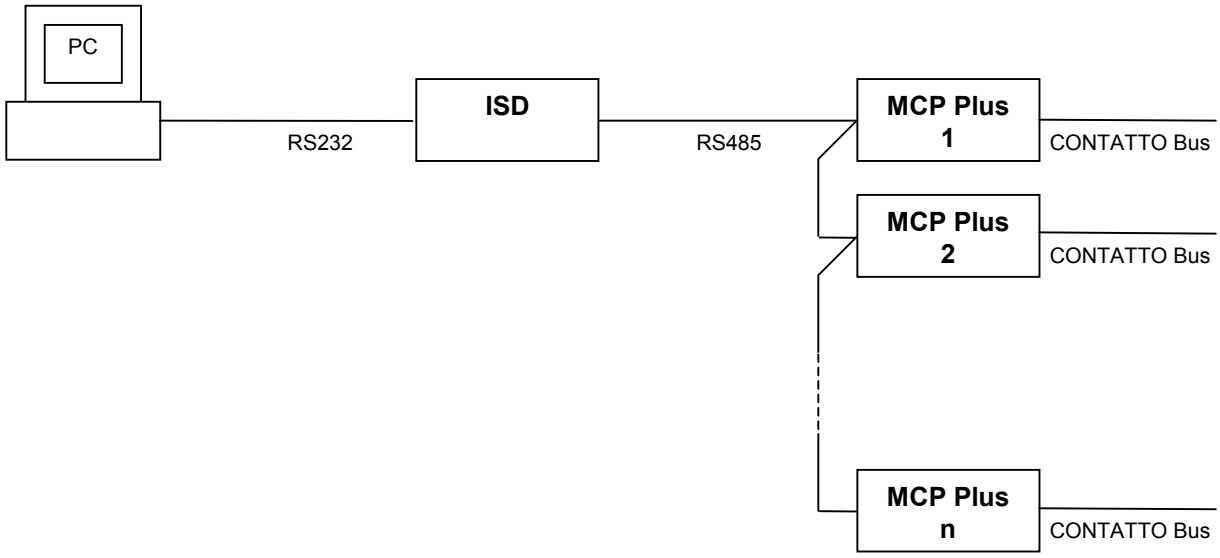
For programming and setting-up of ISD, the software tools **MCP Tools** (**version greater or equal to 3.0**) has to be used (it is the same tools used for MCP). For more details on using this program, see related documentation.

Information between MCPs may be exchanged, through ISD, by means of **virtual points only**; in practice, ISD performs the following functions:

1. It creates a map in its internal RAM that is the "mirror" of all the virtual points of connected MCPs (max **31**) and specified in the ISD program
2. It executes in a **sequential way** the equations specified in the loaded program and, if the result of an equation is different from the previous status, it prepares in a buffer the number of the changed virtual point, its new status and the MCP address to which the new status must be sent
3. It transfer all information in the buffer to the MCPs
4. It restart the polling as specified at point 1

ISD has 2 independent serial ports: one RS485 port to be connected to MCPs and one RS232 port to be connected to a PC (for programming and/or for supervision purposes).

Since all MCP controllers are connected in a RS485 network, it is mandatory **use MCP Plus because they have a RS485 integrated port**. In the following pages, MCP Plus will be named simply MCP. Each MCP must have a different address in the same RS485 network; to do this, the ADDRESS function has to be used (see MCP user's manual).
The following figure shows the block diagram of the connection between ISD, MCPs and PC.



The PC is strictly required only for the programming and setup of MCP controller and ISD interface; Using proper programs, it is possible to use the PC also for programming purposes.

Each MCP controls, according to the program loaded in MCP itself, its own field bus; ISD interface allows to transfer the status of the virtual points from one MCP to another. For example it is possible to control a lamp connected to an output module of MCP2 from a switch connected to an input module of MCP1.
Even if it is possible to connect up to 31 MCPs in the same network, take in account that **the time required by ISD interface for reading all virtual points of all MCPs increases when increasing the number of connected controllers**; as consequence, the input to output delay (response time) increase. This is a normal in a RS485 network, because the information is transferred in serial format.
It is very difficult to exactly evaluate the response time as function of the number of connected MCPs, because this value depends on many parameters; this time may be however approximated by the following formula:

Tr = (number of connected MCPs) x 0.5 seconds

This formula is valid when the communication speed between ISD and MCPs is **19200 Baud** (that is the max value) and when the supervisor does not poll ISD through the RS232 port (see block diagram in the previous page); in facts, the response time increases if a supervisor polls continuously ISD interface.

In the following pages of this manual, the basic elements for connection and programming of ISD interface will be described; it is assumed that the reader know the programming of MCP controllers.

1.1- General description

ISD interface of CONTATTO bus family provides, in its RAM memory, **1000 virtual points** plus the "mirror" of the 1000 virtual points of each connected MCP.
As said above, ISD can exchange the status of virtual point **only** among the connected MCPs; in order to distinguish a virtual point of a MCP from a virtual point of another MCP and from the virtual points of ISD, the following notation must be used:

`Vm.n`

where **m** is the address of MCP and **n** is the number of the virtual point; if m=0, then it is a virtual point of ISD. More precisely:

- Virtual points from V0.1 to V0.1000 are the points of ISD itself
- Virtual points from V1.1 a V1.1000 are those related to MCP which address is 1
- Virtual points from V2.1 a V2.1000 are those related to MCP which address is 2
- ……
- Virtual points from Vn.1 a Vn.1000 are those related to MCP which address is n

Some points of ISD are reserved for specific functions as here bottom described:

- **V0.1000**: read-only point; it is activated when ISD detect the failure of one or more MCPs (generally when one or more MCPs do not answer to the polling of ISD). This point follows the status of the LED MCP.F on the ISD front panel
- **V0.999**: read/write point; this point controls the status of the internal relay of ISD; more precisely, when V0.999 is zero, the relay is energized, when V0.999 is 1 the relay is de-energized
- **V0.998**: read-only point; this points is activated at the end of initialization procedure at the power up or after a re-programming
- **V0.997**: read-only point; this point change its status every 0.5 seconds
- **V0.996**: reserved
- **V0.995**: reserved
- **V0.994**: reserved
- **V0.993**: reserved

2- EQUATIONS: TYPES AND SYNTAX

The equations that can be loaded in ISD interface may be made by logic operators **only**. Logic equations allow to control the status of a virtual point (used as output) by a combination of some virtual points (used as inputs). There is no limit to the amount of inputs that may be combined in a single equation.
The syntax of logic equations for ISD is the following:

`Vm.n = f( Vj.k )`

where **`Vm.n`** is the virtual point **`n`** of MCP which address is **`m`** and **`f( Vj.k )`** is the combination of the points controlling **`Vm.n`**. The notations **`m.n`** and **`j.k`** (or similar ones used in the following pages) are therefore made by the MCP address (**`m`** and **`j`**) and the number of the related virtual point (**`m`** e **`k`**). As said above, **`m`** or **`j`** equal to zero, means that considered virtual point is a point of ISD itself.

The function defining a virtual output may be made by one or more virtual inputs linked together by AND operator (symbol **`&`**) and by OR operator (symbol |); it is possible to complement the logic of an input placing before it the symbol **`!`** (NOT operator).
The **`&`** operator is equivalent, in the electro-mechanical notation, to the series connection of contacts, while the | operator is equivalent to the parallel connection. The function of the NOT operator is equivalent to replace a normally open contact with a normally closed contact.

The priority of **`&`** and | operators in the same equation is the following:

1. **`&`** operator
2. | operator

This priority may be modified using the brackets **`(`** and).

To avoid misunderstandings, follow this rules:

- Virtual points used as **inputs in one MCP** (and therefore MCP will have equations of the type **`Ox.y = Vn`**) **will be used as output by ISD** (and therefore this points will be placed on the left side of symbol **=** in the ISD equation)
- Virtual points used as **outputs in one MCP** (and therefore MCP will have equations of the type **`Vn = Ix.y`**) **will be used as input by ISD** (and therefore this points will be placed on the right side of symbol **=** in the ISD equation)

The following examples show some allowed equations for ISD and the related equation for the MCP controller.

Example 1:

An installation has two MCP controllers (that will be named MCP1 and MCP2); one output of MCP1 bus (`O1.1`) must be controlled by a switch (`I1.1`) connected to the bus of MCP2. Both MCP are connected, by the RS485 network, to ISD interface.

The program to be loaded into **MCP2** will be the following:

```
ADDRESS = 2  // Assign address 2 to MCP
V1 = I1.1    // V1 follows the status of I1.1
```

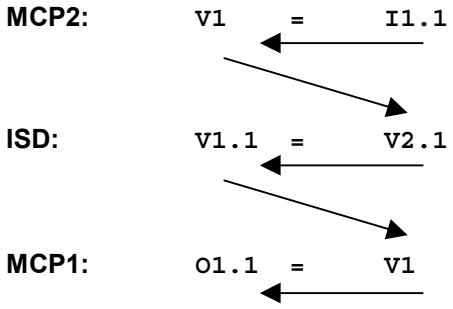
The program to be loaded into **ISD** will be the following:

```
V1.1 = V2.1  // Virtual point 1 of MCP1 follows the virtual point 1 of MCP2
```

The program to be loaded into **MCP1** will be the folowing:

```
ADDRESS = 1  // Assign address 1 to MCP
O1.1 = V1    // O1.1 follows the virtual point V1, or, in other words,
             // O1.1 of MCP1 follows I1.1 of MCP2
```

The following diagram shows the logic way from `I1.1` to `O1.1`, starting from MCP2 and arriving to MCP1:



Example 2:

An installation has two MCP controllers (that will be named MCP1 and MCP2); one output (`O1.1`) of MCP1 bus must be controlled by a pushbutton (`I1.1`) connected to the bus of MCP2; the output must toggle at each pressing of the pushbutton (TOGGLE function). Both MCP are connected, by the RS485 network, to ISD interface.
This example seems to be similar to the previous one, and therefore one may think to change the equation of MCP1 in the previous example from `O1.1 = V1` to `O1.1 = TV1`; this is correct, but since the response times of the RS485 network may be relatively long, it is possible to lose short pressings of the pushbutton. The best strategy in similar cases is to execute the functions involving pulsed inputs (pushbuttons, temporary contacts, etc.) into the MCP controlling the bus to which those inputs are connected. The equations to be loaded into the MCPs and into the ISD interface will be therefore the following:

Program for **MCP2**:

```
ADDRESS = 2  // Assign address 2 to MCP
V1 = TI1.1   // V1 change its status at each variation OFF to ON of I1.1
```

Program for **ISD**:

```
V1.1 = V2.1  // Virtual point 1 of MCP1 follows the virtual point 1 of MCP2
```

Program for **MCP1**:

```
ADDRESS = 1  // Assign address 1 to MCP
O1.1 = V1    // O1.1 follows the virtual point V1, or, in other words,
             // O1.1 of MCP1 toggle at each variation of I1.1 of MCP2
```

Example 3:

An installation has two MCP controllers (that will be named MCP1 and MCP2); one output (`O1.1`) of MCP1 bus must blink when a contact (`I1.1`) connected to the bus of MCP2 closes. Both MCP are connected, by the RS485 network, to ISD interface. To have a constant blink, it is better to use the V997 of the MCP directly controlling the output.
The equations to be loaded into the MCPs and into the ISD interface will be therefore the following:

Program for **MCP2**:

```
ADDRESS = 2          // Assign address 2 to MCP
V1 = I1.1            // V1 follows I1.1
```

Program for **ISD**:

```
V1.1 = V2.1          // Virtual point 1 of MCP1 follows the virtual point 1
                     // of MCP2
```

Program for **MCP1**:

```
ADDRESS = 1          // Assign address 1 to MCP
O1.1 = V1 & V997     // O1.1 blinks when V1 is activated, or, in other words,
                     // O1.1 of MCP1 follows V997 when I1.1 of MCP2 is
                     // activated
```

Example 4:

An installation has two MCP controllers (that will be named MCP1 and MCP2) and one ISD interface; the internal relay of ISD must be **de-energized** when the following anomalies occur:

- Failure of at least one module of MCP1 bus
- MCP1 bus failure (short circuit)
- Failure of at least one module of MCP2 bus
- MCP2 bus failure (short circuit)
- Failure of the communication between ISD and at least one MCP

The equation for ISD will be:

```
V0.999 = V0.1000 | V1.1000 | V2.1000 | V1.999 | V2.999
```

In facts:

- V0.999 is the virtual point of ISD controlling the internal relay of ISD (the relay is de-energized when V0.999 is activated)
- V0.1000 is activated when the RS485 communication with at least one MCP fails
- V1.1000 and V2.1000 are the virtual points of the two MCPs and they will be activated when a module failure occurs on the related bus
- V1.999 and V2.999 are the virtual points of the two MCPs and they will be activated when the a bus failure occurs

The activation of one alarm virtual point will cause the activation of V0.999 and this will cause the de-activation of the internal relay of ISD; the alarm warning device (siren or others) must be connected to the normally closed contact of the relay, and this allows activation of the warning device even in the case of a power supply failure (intrinsic safety).

Example 5:

An installation has three MCP controllers (that will be named MCP1, MCP2 and MCP3) and one ISD interface; output `O7.3` of MCP3 must be activated when the output `I38.4` of MCP1 is closed and, at the same time, the input `I46.3` of MCP2 is opened. This example show how the logic combinations inside ISD may be used.

Program for **MCP1**:
```
ADDRESS = 1  // Assign address 1 to MCP1
V51 = I38.4  // V51 follows I38.4
```

Program for **MCP2**:
```
ADDRESS = 2  // Assign address 2 to MCP2
V87 = I46.3  // V87 assume lo stato di I46.3
```

Program for **ISD**:
```
V3.1 = V1.51 & !V2.87  // Virtual point 1 of MCP3 is equal to the AND of I38.4
                       // of MCP1 and the complement of I46.3 of MCP2
```

Program for **MCP3**:
```
ADDRESS = 3  // Assign address 3 to MCP3
O7.3 = V1    // O7.1 follows V1
```

Example 6:

An installation has three MCP controllers (that will be named MCP1, MCP2 and MCP3) and one ISD interface; the following operations must be made:

- Activation of output `O50.1` of MCP1 when at least one of the inputs `I1.1`, `I1.2` and `I1.3` of MCP1 or MCP2 or MCP3 are activated
- Activation of output `O50.1` of MCP2 when at least one of the inputs `I1.1`, `I1.2` and `I1.3` of MCP1 or MCP2 or MCP3 are activated
- Activation of output `O50.1` of MCP3 when at least one of the inputs `I1.1`, `I1.2` and `I1.3` of MCP1 or MCP2 or MCP3 are activated

This last example shows the using of the internal virtual points of ISD.

Program for **MCP1**:
```
ADDRESS = 1  // Assign address 1 to MCP1
V1 = I1.1 | I1.2 | I1.3
O50.1 = V2
```

Program for **MCP2**:
```
ADDRESS = 2  // Assign address 2 to MCP2
V1 = I1.1 | I1.2 | I1.3
O50.1 = V2
```

Program for **MCP3**:
```
ADDRESS = 3  // Assign address 3 to MCP3
V1 = I1.1 | I1.2 | I1.3
O50.1 = V2
```

Program for **ISD**:
```
V0.1 = V1.1 | V2.1 | V3.1
V1.2 = V0.1
V2.2 = V0.1
V3.2 = V0.1
```

3- EQUATION WRITING

The equation writing is the first step of the ISD interface programming. The equations must be written according to the syntax described in the previous paragraphs.

To write the equations, the software package **MCPTools** has to be used; this package is provided by **DUEMMEGI** together to MCP module and ISD interface. MCPTools works on a Personal Computer under WINDOWS® environment and allows an easy writing of the program and system setup. **For more details about the use of this tools, refer to on-line help of the program itself.**

MCPTools essentially includes:

- a text editor to write the program
- a compiler to allow the translation of an ASCII file, containing the operating equations, in a binary file adequate to be transferred in the non volatile memory (FLASH type) of MCP module or ISD interface
- a simulator to verify the written program, or a part of it, before to transfer it into MCP memory (ISD program cannot be simulated)
- an utility to transfer the program from the PC to MCP or ISD (and vice versa)
- a map window to display the status of input and output modules installed in the plant

The file containing the equation is in ASCII format and must have the **.EQU** extension; as example:

***filename*.EQU**

where *filename* is the name of the program file and may be any name allowed by the WINDOWS® syntax.
The **.EQU** extension is **mandatory** because the sequential steps of MCP programming (compiling and transferring) require that the source file have that extension.

ISD interface programming takes place in a 3 sequential steps, through the MCPTools support:

1. building (or editing) of the *filename*.EQU file, containing the operating program
2. compiling of *filename*.EQU, that is the conversion of the ASCII file in the related *filename*.BIN written in a format ready to be transferred into ISD memory
3. uploading of *filename*.BIN into ISD memory

If some syntax errors are detected during the step 2, these ones will be signaled by the compiler, together to some information about the error type and the line number where the error occurs.

Note: the addresses of MCPs to be polled by ISD are automatically deduced by the loaded equations; no additional information are therefore needed for ISD interface.

3.1- Rule for equations writing

Generally, each equation must be written according to the syntax described in its relevant paragraph (logic, counter, timer, etc. ...).

The following rules have to be observed:

- Spaces and TAB characters have no meaning. They will be ignored by the compiler but **the use of some space characters between the terms of an equation are strongly recommended for a best readability of the program**
- An equation can be broken on several lines using the symbol **** (backslash) at the line end to specify that the equation will continue on the next line

- The equation finishes at the end of the line (if the \ symbol is not specified)
- The // symbol (two slashes) declares that the following words, until the line end, are comments, and so they will be ignored by the compiler. **The comments are very useful for best readability and documentation of the program file**. The use of the comment is strongly recommended to describe each equation in the program
- Both upper case and lower case characters can be used for the equation writing

To write and compile a program, the connection between ISD module and the PC is not required.

3.2- Compiling the equations

The compiling is the second step of ISD programming process. **The file containing the written equations (.EQU extension) must be compiled through the proper menu item of MCPTools utility**.

The compiler processes the written equations, check the syntax and the congruence, warns the errors if any and link the data in a binary file which name is the same as the .EQU file but with .BIN extension. The binary file is not in a printable format but it is adequate to be transferred in the MCP memory.

WARNING: ***the programs written for ISD MUST be compiled setting the compiling option of MCPTools to ISD. To do this, from the menu of MCPTools select "Compile" and then "Version ISD".***

To write and compile a program, it is not necessary that MCP module be connected to the PC. If during the compiling process one or more errors occur, they will be displayed on the screen of the PC and the program continues to check all the equations but no binary file will be generated.

The compiler may also reports some WARNINGS: this means that no errors have been detected but there are some points to be verified before to upload the program to ISD memory; the binary file will be however created even if one or more warning messages occur.

3.3- Uploading the program into ISD module

Last step of ISD programming process is the **uploading into its flash MEMORY of the binary file** containing the system configuration and the equations code.
The uploading is made by the proper menu item of MCPTools utility trough the RS232 port of PC connected to the ISD serial port.
Before uploading, be sure to have properly set MCPTools (from the menu of MCPTools select "Compile" and then "Version ISD").

The uploading of the program requires that ISD module be supplied and connected to PC by means of the cable provided with ISD.

Note: ***ISD baud rate factory setting is 19200****; if a slower speed is needed, set the internal jumpers or micro-switches of ISD module as described in next chapter.*

4- SETTING UP

4.1- Connections

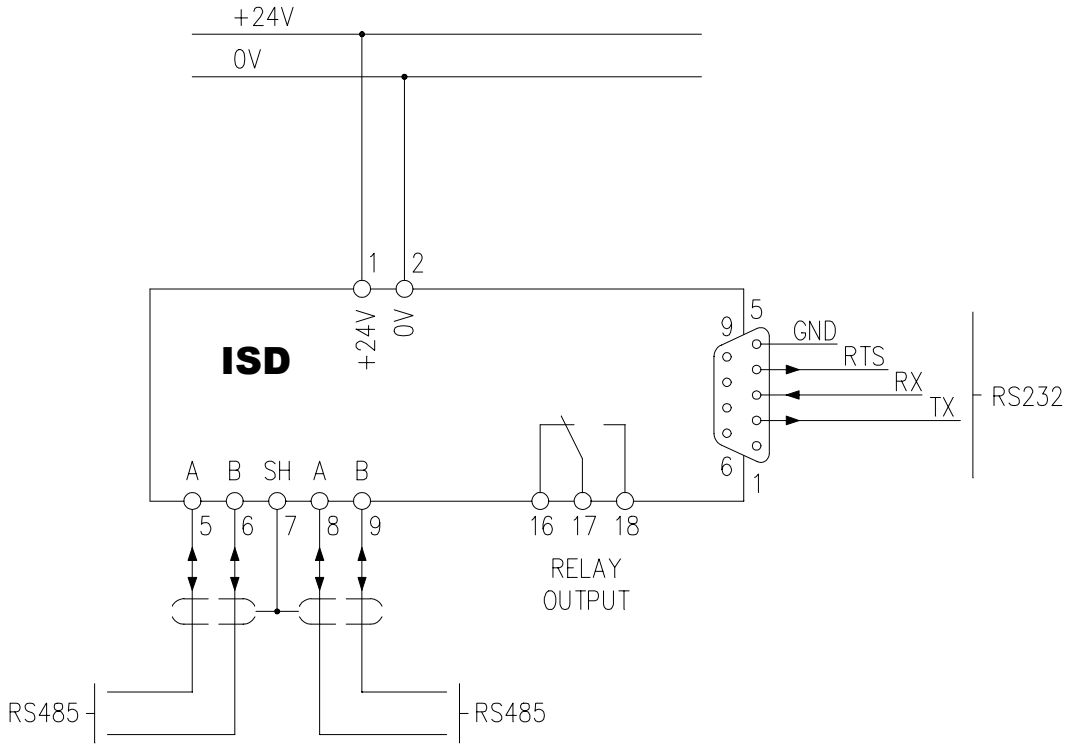
ISD interface is available in DIN modular housing (9 modules size). ISD features a 9-pole removable terminal block for the connection to the 24Vdc power supply and to RS485 network and a 3-pole fixed terminal block internally connected to the contact of the internal relay. As said, this relay is **de-energized when V0.999 is activated**; this relay may be used for signaling, by proper programming, the system anomalies (e.g. fault modules, bus failure, etc.); the contact rating is 1A @ 125Vac.

A serial RS232 port on the front panel allows the connection between ISD module and a Personal Computer; a serial RS485 port, totally independent from the other port, allows the connection to MCP Plus. These ports are **electrically insulated** each one form the other and from the power supply, thanks to internal opto-coupler and dc/dc converters.

RS485 port of ISD is doubled into 4 terminals (plus another terminal for the shield) in order to make easy the multi-drop connection: in other words, terminals 5 and 8 (signal "A") are internally shorted together; in the same way, terminals 6 and 9 (signal "B").

WARNING: as for all RS485 networks, ***radial connections must be avoided***; in addition, RS485 line ***must be loaded, bot at the beginning and at the end, by a 120 Ohm 1/2W resistor*** between terminals A and B. The maximum number of device that can be connected on RS485 line must be limited to 32. The RS485 cable must be shielded and the shield must be connected to all Sh terminals of ISD and MCPs.

Following figure shows the connections to be made.



4.3- Baud rate selection

Baud rate factory setting of the serial ports of ISD is 19200; if for any reasons this speed has to be changed, proceed as here bottom described. Available baud rates are:

- 19200 Baud
- 9600 Baud
- 4800 Baud
- 2400 Baud

To change the setting, remove the cover between the RS485 terminal block and the alarm contact terminal block; set the dip-switches to the required baud rate according to the truth table printed on the board itself.

To make operating the new baud rate selection, switch-off and then switch-on again ISD module.

WARNING: the baud rate setting of RS232 and RS485 port is always the same.

5- DIAGNOSTICS

5.1- Diagnostics of CONTATTO ISD interface

ISD interface provides an alarm warning through one red LED, named MCP.F(MCP fail), on the front panel; this LED is lighted when ISD cannot communicate (for a fixed timeout) with one or more MCPs connected to the RS485 network. When the communication is restored, the LED MCP.F is automatically switched off.

To know what MCP failed, MCPTools may be used; select from the menu "Compile" and then "Version ISD", then select "Supervision" and Show Map". At this point, if the communication between the PC and ISD is enabled, the map will show the MCP controller related to the program currently loaded into ISD interface. The graphical representation in the map is in red color for the fault MCPs and in green color for the properly working MCPs. For more details, refer to the on-line help of MCPTools program.

Two yellow LEDs (named TX) and two red LEDs (named RX) on the front panel of ISD interface monitor the communication status on RS232 and RS485 ports.

6- TECHNICAL CHARACTERISTICS

Power supply voltage	24Vdc ± 25%
Max current consumption	100mA
Alarm contact rating	1A@125Vac or 60Vdc (resistive load)
Number of employed processors	1
Reading time for each MCP (TYP)	150msec (@ 19200 Baud)
Time needed for changing a virtual point in one MCP (TYP)	30msec (@ 19200 Baud)
User program memory size	FLASH type 128 Kbytes
RAM Memory size	32 Kbytes
Internal allowable virtual points	1000
Serial ports	RS232 and RS485 (electrically insulated)

7- OUTLINE DIMENSIONS