1 ORION ITALIA SERIES MODBUS PROTOCOL.

The ORION ITALIA SERIES implement a subset of the AEG Modicon Modbus serial communication standard. Many devices support this protocol directly with suitable interface card, allowing direct connection of relays. The Modbus protocol is hardware-independent; that is, the physical layer can be any of variety of standard hardware configurations. This includes RS232, RS422, RS485, fibber optics, etc. The ORION ITALIA RELAYS include a rear terminal RS422/RS485 port. Modbus is a single master multiple slave protocol suitable for a multi-drop configuration as provided by RS422/RS485 hardware. In this configuration up to 32 slaves can be daisy-chained together on a single communication channel.

The ORION ITALIA SERIES is always a Modbus slave. It cannot be programmed as a Modbus master. The Modbus protocol exists in two versions: Remote Terminal Unit (RTU, binary) and ASCII. Only the RTU version is supported by the Orion Italia Relay. Monitoring, programming and control functions are possible using read and write register commands.

2 ELECTRICAL INTERFACE.

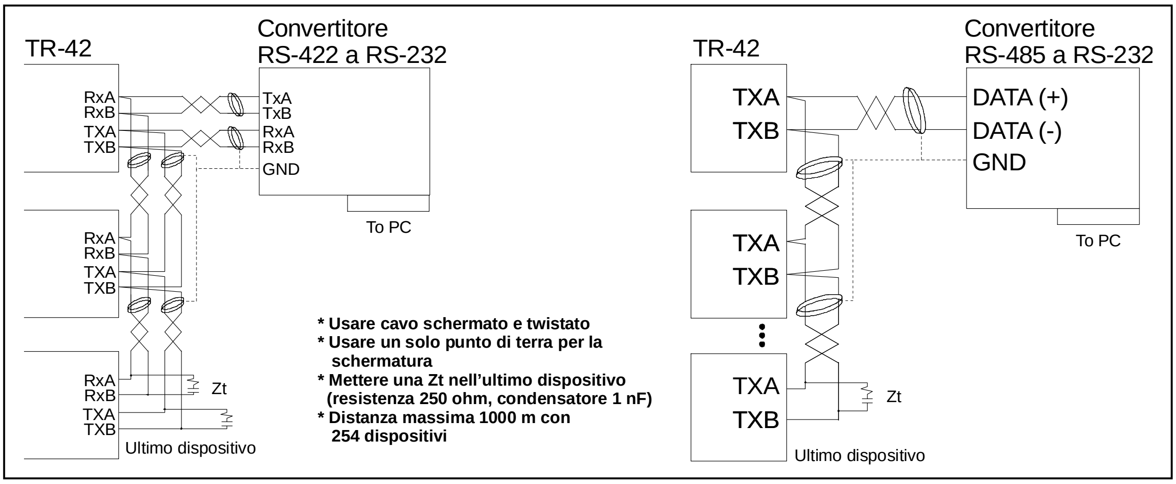
The hardware or electrical interface is any of the following:

a. Two-wire RS485 for the rear terminal connector

b. Four-wire RS422 for the rear terminal connector.

In a two-wire RS485 link, data flow is bi-directional. The four-wire RS422 port uses two terminals for receive lines, and two other terminals for transmit lines. As a result, both RS422 and RS485 ports cannot be used at the same time. The data flow is half duplex. That is, data is never transmitted and received at the same time.

RS485 and RS422 lines should be connected in a daisy chain configuration (avoid star connections) with terminating resistors and capacitors installed each end of the link, i.e. at the master end and the slave farthest from the master. That value of the terminating resistors should be equal to the characteristic impedance of the line. This is approximately 120 Ohms for standard 24 AWG twisted pair wire. The value of the capacitors should be 1 nF. Shielded wire should always be used to minimize noise. Polarity is important in RS485 communications. See figure below for more details.



3 DATA FRAME FORMAT AND DATA RATE.

One data frame of an asynchronous transmission to or from a Orion Italia Relay consists of 1 start bit, 8 data bits, not parity and 1 stop bit. This produces a 10 bit frame. This is important for transmission though modems at high bit rates (11 bit data frames are not supported by hayes modems at bit rates of greater than 300 bps).

The rear RS485/RS422 external communication port of the Orion Italia Relay supports operation at 1200,2400,4800 and 9600 baud.

4 DATA PACKET FORMAT.

A complete request/response consists of the following bytes transmitted as separate data frames:

Master Query Message:

SLAVE ADDRESS	(1 byte)
FUNCTION CODE	(1 byte)
DATA	(variable number of bytes depending on FUNCTION CODE)
CRC	(2 bytes)

Slave Response Message

SLAVE ADDRESS	(1 byte)
FUNCTION CODE	(1 byte)
DATA	(variable number of bytes depending on FUNCTION CODE)
CRC	(2 bytes)

A message is terminated when no data is received for a period of 3 1/2 character transmission times. Consequently, the transmitting device must not allow gaps between bytes larger than this interval (about 3ms at 9600 baud).

SLAVE ADDRESS: This is the first byte of every message. This byte represents the user-assigned address of the slave device that is to receive the message sent by the master. Each slave device must be assigned a unique address, and only the addressed slave will respond to a message that starts with its address. In a master query message the SLAVE ADDRESS represents the address of the slave to which the request is being sent. In a slave response message the SLAVE ADDRESS is a confirmation representing the address of the slave that is sending the response. A master query message with a SLAVE ADDRESS of 0 indicates a broadcast command. All slaves on the communication link will take action based on the message, but none will respond to the master.

FUNCTION CODE: This is the second byte of every message. Modbus defines function codes of 1 to 127. The Orion Italia Relay implements some of this functions. See section 7 for details of the function codes supported by the Orion Italia Relay Series Modbus Protocol. In a master query message the FUNCTIONS CODE tells the slave what action to perform. In a slave response message, if the FUNCTION CODE sent from the slave is the same as the FUNCTION CODE sent from the master then the slave performed the function as requested.

DATA: This will be a variable number of bytes on the FUNCTION CODE. This may include actual values, setpoints or addresses sent by the master to the slave or by the slave to the master. See section 7 for a description of the Orion-supported functions and the data required for each.

CRC: This is a two byte error checking code. CRC is sent LSByte first followed by the MSByte. The RTU version of Modbus includes a two byte CRC-16 (16 bit cyclic redundancy check) with every message. The CRC-16 algorithm essentially treats the entire data stream (data bits only; start, stop and parity ignored) as one continuous binary number. This number is first shifted left 16 bits and then divided by a characteristic polynomial (11000000000000101B). The 16 bit remainder of the division is appended to the end of the message, MSByte first. The resulting message including CRC, when divided by the same polynomial at the receiver will give a zero remainder if no transmission errors have occurred. If a Orion Modbus slave device receives a message in which an error is indicated by the CRC-16 calculation, the slave device will not respond to the message. A CRC-16 error indicates that one or more bytes of the message were received incorrectly and thus the entire message should be ignored in order to avoid the slave device performing any incorrect operation. The CRC-16 calculation is an industry standard method used for error detection. An algorithm is included in section 5 CRC-16 algorithm to assist programmers in situations where no standard CRC-16 calculation routines are available.

5 CRC-16 ALGORITHM

Once the following algorithm is completed, the working register "A" will contain the CRC value to be transmitted. Note that this algorithm requires the characteristic polynomial to be reverse bit ordered. The most significant bit of the characteristic polynomial is dropped, since it does not affect the value of the remainder. The following symbols are used in the algorithm:

Symbols:

-->	data transfer
A	16 bit working register
Alow	low order byte of A
Ahigh	high order byte of A
CRC	16 bit CRC-16 value
i,j	loop counter
(+)	logical EXCLUSIVE-OR operator
N	total number of data bytes
Di	i-th data byte (i=0 to N-1)
G	16 bit characteristic polynomial =1010000000000001(binary) with MSbit dropped and bit order reversed
shr(x)	right shit operator (the LSbit of x is shifted into a carry lag, a '0' is shifted into the MSbit of x, all other bits are shifted right one location)

Algorithm:

1.	**FFFF(hex) --> A**	
2.	**O --> i**	
3.	**O --> j**	
4.	**Di (+) Alow --> Alow**	
5.	**j + 1 --> j**	
6.	**shr (A)**	
7.	**Is there a carry ?**	**No: go to step 8**
		Yes: G (+) A --> A and continue
8.	**Is j = 8 ?**	**No: go to 5**
		Yes: continue
9.	**i + 1 --> i**	
10.	**Is i = N ?**	**No: go to 3**
		Yes: continue
11.	**A ----> CRC**	

6 MESSAGE TIMING

Communication message synchronization is maintained by timing constraints. The receiving device must measure the time between the reception of characters. If three and one half character times elapse without a new character or completion of the message, then the communication link must be reset (i.e. all slaves start listening for a new query message from the master). Thus at 1200 baud a delay of greater than 3.5 x 1/1200 x 10 = 29.2 msec cause the communication link to be reset. At 9600 baud a delay of greater than 3.5 x 1/9600 x 10 = 3.6 ms will cause the communication link to be reset. Most master query messages will be responded to in less than 50 ms. The maximum time for the Orion Italia Relay to return a slave response message for any function code will never exceed 1 second.

7 SUPPORTED FUNTION CODES

The second byte of every message is the function code. Modbus defines function codes of 01h to 7Fh. The Orion Italia Relay Modbus protocol supports some of these functions, as summarized in Table No. 1

TABLE No. 1

FUNCTION CODE MODBUS PROT. (HEX)	FUNCTION CODE ORION ITALIA (HEX)	DEFINITION
03	03	READ SETPOINTS or ACTUAL VALUES
04	04	READ SETPOINTS or ACTUAL VALUES
05	05	EXECUTE OPERATION
06	06	STORE SINGLE SETPOINTS
10	10	STORE MULTIPLES SETPOINTS

Since some programmable logic controllers only support function codes 03h (or 04h) and 10h, most of the above Modbus commands can be performed by reading from or writing to special addresses in the Orion Italia Relay memory map using these function codes.

7.1 FUNCTION CODE 03H or 04H - READ SETPOINTS OR ACTUAL VALUES.

Modbus implementation: Read Holding Registers

Orion Italia Relay implementation: Read Actual Values or Setpoint

The Orion Italia Relay implementation of Modbus views "holding registers" as any setpoint or actual values register in the Orion Italia Relay memory map. Registers are 16 (two byte) values transmitted high order byte first. Thus all Orion Italia Relay setpoints and actual values in the memory map are sent as two byte registers. This function code allows the master to read one or more consecutive setpoints or actual values from addressed slave device.

The slave response to these function codes is the slave address, function code, a count of the number of data bytes to follow, the data itself and the CRC. Each data item is sent as a two byte number with the high order byte sent first. The CRC is sent as a two byte number with the low order byte sent first.

The maximum number of values of Setpoints that can be read in a single message is 6 word (12 bytes). The TR-42 Setpoint data starts at address 0100h.

The maximum number of values of Actual Values that can be read in a single message is 14 word (28 bytes). The TR-42 Actual Values data starts at address 0200h.

MESSAGE FORMAT EXAMPLE:

Request to read 4 register values starting address 0200h from slave device 1.

Master query message	Example(hex)	
SLAVE ADDRESS	01	query message for slave 01 = 01h
FUNCTION CODE	04	read Actual Values
DATA STARTING ADDRESS-high order	02	data starting at address FB2Ah
DATA STARTING ADDRESS-low order byte	00	
NUMBER OF REGISTERS-high order byte	00	4 register value = 4 word total
NUMBER OF REGISTER-low order byte	04	
CRC-low order byte	F0	CRC calculated by the master
CRC-high order byte	71	

If the function code or the address of any of the requested data is illegal, the slave will not respond the message. Otherwise, the slave will respond as follows:

Slave response message	Example (hex)	
SLAVE ADDRESS	01	response message from slave 1 = 01h
FUNCTION CODE	04	read Actual Values
BYTE COUNT	08	4 register values = 8 bytes total
DATA #1-high order byte	00	register value in address 0200= 003Ah
DATA #1-low order byte	3A	
DATA #2-high order byte	00	register value in address 0201=003Dh
DATA #2-low order byte	3D	
DATA #3-high order byte	00	register value in address 0202=0039h
DATA #3-low order byte	39	
DATA #4-high order byte	00	register value in address 0203=001Bh
DATA #4-low order byte	1B	
CRC-low order byte	43	CRC calculated by the slave
CRC-high order byte	CD	

7.2 FUNCTION CODE 05H - EXECUTE OPERATION

Modbus implementation: Force Single Coil

Orion Italia Relay implementation : Execute Operation

This function code allows the master to request TR-42 to perform specific command operations. The commands Number Listed in the Commands Area of the Relay memory map correspond to operations codes for function code 05h.

The Slave Response to this function is to echo the entire master transmission.

The operation codes can also be initiated by writing to the commands area of the memory map using function codes 06h or 10h.

MESSAGE FORMAT EXAMPLE:
Request to Clear the Maximum Temperatures Registered in TR-42 in slave device 1.

Master query message	Example(hex)	
SLAVE ADDRESS	01	Query message for slave 01 = 01h
FUNCTION CODE	05	Execute Operation
DATA STARTING ADDRESS-high order	00	Clear Maximum Temperatures Command
DATA STARTING ADDRESS-low order byte	10	
NUMBER OF REGISTERS-high order byte	FF	Perform Function
NUMBER OF REGISTER-low order byte	00	
CRC-low order byte	8D	CRC calculated by the master
CRC-high order byte	FF	

Slave response message	Example (hex)	
SLAVE ADDRESS	01	Message from slave 01 = 01h
FUNCTION CODE	05	Execute Operation
DATA STARTING ADDRESS-high order	00	Clear Maximum Temperatures Command
DATA STARTING ADDRESS-low order byte	10	
NUMBER OF REGISTERS-high order byte	FF	Perform Function
NUMBER OF REGISTER-low order byte	00	
CRC-low order byte	8D	CRC calculated by theSlave
CRC-high order byte	FF	

7.3 FUNCTION CODE 06H - STORE SINGLE SETPOINTS

Modbus implementation: Preset Single Register
Orion Italia Relay implementation : Store Single Setpoints

This function code allows the master to store single setpoints into the memory map of the TR-42.
The Slave Response to this function is to echo the entire master transmission.

MESSAGE FORMAT EXAMPLE:
Request slave device 01h to write the value 0064h at setpoint address 0100h

Master query message	Example(hex)	
SLAVE ADDRESS	01	query message for slave 01 = 01h
FUNCTION CODE	06	Store Single Setpoints
DATA STARTING ADDRESS-high order	01	Setpoint Address 0100h
DATA STARTING ADDRESS-low order byte	00	
NUMBER OF REGISTERS-high order byte	00	Data for Address 0100h = 0064h
NUMBER OF REGISTER-low order byte	64	
CRC-low order byte	89	CRC calculated by the master
CRC-high order byte	DD	

Slave response message	Example (hex)	
SLAVE ADDRESS	01	Message from slave 01 = 01h
FUNCTION CODE	06	Store Single Setpoints
DATA STARTING ADDRESS-high order	01	Setpoint Address 0100h
DATA STARTING ADDRESS-low order byte	00	
NUMBER OF REGISTERS-high order byte	00	Data Stored in Address 0100h = 0064h
NUMBER OF REGISTER-low order byte	64	
CRC-low order byte	89	CRC calculated by the Slave
CRC-high order byte	DD	

7.4 FUNCTION CODE 10H -STORE MULTIPLE SETPOINTS

Modbus implementation: Preset Multiple Register
Orion Italia Relay implementation : Store Multiple Setpoints

This function code allows the master to modify the contest of a one or more consecutive setpoint in the addressed slave device. Setpoint registers are 16 bit (two byte) values transmitted high order byte first.

The maximum number of register values (setpoints) that can be stored in a single message is 6 word (12 bytes). The TR-42 Setpoint data starts at address 0100h.

To store the value of one or more setpoints in the internal memory of the TR-42, the following steps shall be realized:
a) First shall be read all setpoint group (6 word = 12 bytes) with function 03h or 04h.
b) Modify the values of setpoints according to setpoint map.
c) Send all setpoint group, 6 word (12 bytes) with function 10h.

The TR-42 response to this function code is to echo the slave address, function code, starting address, the number of setpoints stored, and the CRC.

MESSAGE FORMAT AND EXAMPLE:

Request slave device 11h to write the value 0064h at setpoint address 0100h, and the value 0070h at setpoint address 0101h.

Master query message	Example (hex)
SLAVE ADDRESS	11 query for slave 11h
FUNCTION CODE	10 store multiple setpoint values
DATA STARTING ADDRESS-high order byte	01 data starting at address 0100
DATA STARTING ADDRESS-low order byte	00
NUMBER OF SETPOINTS-high order byte	00 2 setpoint values = 2 word
NUMBER OF SETPOINTS-low order byte	02
BYTE COUNT	04 4 byte of data
DATA #1-high order byte	00 data for address 0100h=0064h

DATA #1-low order byte	64
DATA #2-high order byte	00 data for address 0101h=0070h
DATA #2-low order byte	70
CRC-low order byte	EA CRC calculated by the master
CRC-high order byte	C4

If the function code or the address or value of any of the data, is illegal, the slave will not respond to the message. Otherwise, the slave will respond as follows:

Master query message	Example (hex)
SLAVE ADDRESS	11 Message from slave 11h
FUNCTION CODE	10 store multiple setpoint values
DATA STARTING ADDRESS-high order byte	01 data starting at address 1100h
DATA STARTING ADDRESS-low order byte	00
NUMBER OF SETPOINTS-high order byte	00 2 setpoint values = 2 word
NUMBER OF SETPOINTS-low order byte	02
CRC-low order byte	42 CRC calculated by the slave
CRC-high order byte	A4

8 MEMORY MAP INFORMATION

The data stored in the TR-42 is grouped as Setpoints, Actual Values, Product ID & Commands. Setpoints can be read and written by a master computer. Actual Values & Product ID are read only, Commands are Write Only. All setpoints and Actual Values are stored as two bytes values. Addresses are listed in hexadecimal. Data values (Setpoint ranges, increments, factory value) are in decimal.

MEMORY MAP

TR-42 Relay – Model 1 – Software Versions:from v1.05 to v1.11

Add (Hex)	Type	Size (word)	Description	Unit	Range	Step	Initial Value	Format	Read/ Write
0000	**Product ID**	1	Product Code	-	-	-	2	F1	R
0001		1	Product Model	-	-	-	1	F1	R
0002		1	Version Number	-	-	-	1.00	F2	R
0080	**Commands**	1	Commands Operations & PushButtons Simulation	-	-	-	-	F3	W
0100	**Setpoints**	1	L1 Value	℃	0~220	1	-	F1	R/W
0101		1	L2 Value	℃	0~220	1	-	F1	R/W
0102		1	FAN Low Value	℃	0~220	1	-	F1	R/W
0103		1	FAN High Value	℃	0~220	1	-	F1	R/W
0104		1	General Configurations	-	-	-	-	F4	R/W
0105		1	BAUDRATE	-	0-3	1	3	F5	R/W
0200	**Actual**	1	RTD Input 1	℃	(-10)~(230)	1	-	F1	R
0201	**Values**	1	RTD Input 2	℃	(-10)~(230)	1	-	F1	R
0202		1	RTD Input 3	℃	(-10)~(230)	1	-	F1	R
0203		1	RTD Input 4	℃	(-10)~(230)	1	-	F1	R
0204		1	Maximun RTD Input 1	℃	(-10)~(230)	1	-	F1	R
0205		1	Maximun RTD Input 2	℃	(-10)~(230)	1	-	F1	R
0206		1	Maximun RTD Input 3	℃	(-10)~(230)	1	-	F1	R
0207		1	Maximun RTD Input 4	℃	(-10)~(230)	1	-	F1	R
0208		2	Display Values	-	-	-	-	F6	R
020A		1	Leds	-	-	-	-	F7	R
020B		1	Blink Status	-	-	-	-	F8	R
020C		1	Relay Status	-	-	-	-	F9	R
020D		1	General Status	-	-	-	-	F10	R

TR-42 DATA FORMATS

Format Code	Type	Value	Definition
F1	**Integer**		**Signed Integer Value**
			Example: 123 saved as 123
F2	**Integer**		**Signed Integer Value with 2 decimals**
			Example: 1.00 saved as 100
F3	**Integer**		**Commands Operations Codes**
		00 hex	No Operación
		01 hex	PC Control Flag **Activation** (This Flag will deactivate in 1 minute)
		02 hex	PC Control Flag **Deactivation**
		03 hex	**UP** PushButton Simulation
		04 hex	**DOWN** PushButton Simulation
		05 hex	**ENTER** PushButton Simulation
		06 hex	**MODE** PushButton Simulation
		10 hex	Maximun Temperature **CLEAR**
F4	**16 Bits BitMap**		**General Configurations Bits**
		Bit 0	0 = Three (3) RTD inputs Connected
			1 = Four (4) RTD inputs Connected
		Bit 1	0 = Fan not Instaled
			1 = Fan Instaled
		Bit 2	0 = Three (3) RTD inputs for Command Fan
			1 = Four (4) RTD inputs for Command Fan
		Bit 3	0 = weekly Automatic ON Fan **DEACTIVATED**
			1 = weekly Automatic ON Fan **ACTIVATED**
		Bit 4	0 = **RS-422** port configurated
			1 = **RS-485** port configurated
		Bit 5 ~ Bit 7	Not Used
		Bit 8 ~ Bit 15	Slave Address (Range from 1 to 255)
F5	**Integer**		**BaudRate Configuration**
		00 hex	1200 Bps
		01 hex	2400 Bps
		02 hex	4800 Bps
		03 hex	9600 Bps
F6	**4xASCII**		**ASCII Values for Display Values**
		Byte #1	ASCII Value for Hundreds Digit
		Byte #2	ASCII Value for Tens Digit
		Byte #3	ASCII Value for Unit Digit
		Byte #4	ASCII Value for Individual Digit

TR-42 DATA FORMATS

F7	16 Bits BitMap		LEDS Status Bits
		Bit 0	0 = **PROG** Led **OFF**
			1 = **PROG** Led **ON**
		Bit 1	0 = **°C MAX** Led **OFF**
			1 = **°C MAX** Led **ON**
		Bit 2	0 = **AUTO SCAN** Led **OFF**
			1 = **AUTO SCAN** Led **ON**
		Bit 3	0 = **L1** Led **OFF**
			1 = **L1** Led **ON**
		Bit 4	0 = **L2** Led **OFF**
			1 = **L2** Led **ON**
		Bit 5	0 = **FAN** Led **OFF**
			1 = **FAN** Led **ON**
		Bit 6	0 = **FAULT** Led **OFF**
			1 = **FAULT** Led **ON**
		Bit 7	0 = **IN 1** Led **OFF**
			1 = **IN 1** Led **ON**
		Bit 8	0 = **IN 2** Led **OFF**
			1 = **IN 2** Led **ON**
		Bit 9	0 = **IN 3** Led **OFF**
			1 = **IN 3** Led **ON**
		Bit 10	0 = **IN 4** Led **OFF**
			1 = **IN 4** Led **ON**
		Bit 11 ~ Bit 15	Not Used

TR-42 DATA FORMATS

F8	16 Bits BitMap		LEDS & DISPLAY Blink Status Bits
		Bit 0	0 = **PROG** Led **NOT BLINKING**
			1 = **PROG** Led **BLINKING**
		Bit 1	0 = **°C MAX** Led **NOT BLINKING**
			1 = **°C MAX** Led **BLINKING**
		Bit 2	0 = **AUTO SCAN** Led **NOT BLINKING**
			1 = **AUTO SCAN** Led **BLINKING**
		Bit 3	0 = **L1** Led **NOT BLINKING**
			1 = **L1** Led **BLINKING**
		Bit 4	0 = **L2** Led **NOT BLINKING**
			1 = **L2** Led **BLINKING**
		Bit 5	0 = **FAN** Led **NOT BLINKING**
			1 = **FAN** Led **BLINKING**
		Bit 6	0 = **FAULT** Led **NOT BLINKING**
			1 = **FAULT** Led **BLINKING**
		Bit 7	0 = **IN 1** Led **NOT BLINKING**
			1 = **IN 1** Led **BLINKING**
		Bit 8	0 = **IN 2** Led **NOT BLINKING**
			1 = **IN 2** Led **BLINKING**
		Bit 9	0 = **IN 3** Led **NOT BLINKING**
			1 = **IN 3** Led **BLINKING**
		Bit 10	0 = **IN 4** Led **NOT BLINKING**
			1 = **IN 4** Led **BLINKING**
		Bit 11	Not Used
		Bit 12	0 = **Hundreds** Digit **NOT BLINKING**
			1 = **Hundreds** Digit **BLINKING**
		Bit 13	0 = **Tens** Digit **NOT BLINKING**
			1 = **Tens** Digit **BLINKING**
		Bit 14	0 = **Unit** Digit **NOT BLINKING**
			1 = **Unit** Digit **BLINKING**
		Bit 15	0 = **Individual** Digit **NOT BLINKING**
			1 = **Individual** Digit **BLINKING**

TR-42 DATA FORMATS

F9	16 Bits BitMap		Output Relay Status Bits
		Bit 0	0 = **L1** Realy **DE-ENERGIZED**
			1 = **L1** Realy **ENERGIZED**
		Bit 1	0 = **L2** Realy **DE-ENERGIZED**
			1 = **L2** Realy **ENERGIZED**
		Bit 2	0 = **FAULT** Realy **DE-ENERGIZED**
			1 = **FAULT** Realy **ENERGIZED**
		Bit 3	0 = **FAN** Realy **DE-ENERGIZED**
			1 = **FAN** Realy **ENERGIZED**
		Bit 4 ~ Bit 15	NoT Used
F10	**16 Bits BitMap**		**General Status Bits**
		Bit 0	0 = **L1 Normal**
			1 = **L1 Active**
		Bit 1	0 = **L2 Normal**
			1 = **L2 Active**
		Bit 2	0 = **FAN Normal**
			1 = **FAN Active**
		Bit 3	0 = **FAULT Normal**
			1 = **FAULT Active**
		Bit 4	0 = **FAN WEEKLY Normal**
			1 = **FAN WEEKLY Active**
		Bit 5 ~ Bit 7	NoT Used
		Bit 8	0 = **INPUT RTD 1 Normal**
			1 = **INPUT RTD 1 Failure**
		Bit 9	0 = **INPUT RTD 2 Normal**
			1 = **INPUT RTD 2 Failure**
		Bit 10	0 = **INPUT RTD 3 Normal**
			1 = **INPUT RTD 3 Failure**
		Bit 11	0 = **INPUT RTD 4 Normal**
			1 = **INPUT RTD 4 Failure**
		Bit 12 ~ Bit 15	NoT Used