ENCLOSURE 2

RADIATION MONITORING SYSTEM

PROTOCOL TECHNICAL SPECIFICATIONS

VPL No.: 5012 3-606-1-9-0

- ☐ 1. APPROVED – Mfg. may proceed.
- ☐ 2. APPROVED EXCEPT AS NOTED – Make changes and resubmit. Mfg. may proceed.
- ☐ 3. NOT APPROVED – Correct and resubmit for review. Not to be used for field use.
- ■ 4. REFERENCE DOCUMENT – "Information Only"

Quality Class: II

SOUTHERN CALIFORNIA EDISON COMPANY

Drawings are reviewed and approved for arrangements and conformance to specification only. Approval does not relieve the submitter from the responsibility of adequacy and suitability of design, materials, and/or equipment represented. 5012 3-606-1-12 R2

Quality Classification Ref. Doc. No. 5012-606-1 R0

* DM o. OTHER as required by DRADM

INITIALED	
SQE	DATE 7-31-95
QS	DATE 7-31-95
PRE	DATE
DM*	DATE
OTHER*	DATE

SCE 57-939 REV 0 5/93 (REFERENCE SO123-XXIV-57.8.28)

PGD-00082
Revision 0
July 6, 1995

PGD00082.DOC

RMS Protocol Technical

Specifications

RECEIVED CDM

AUG 0 4 1995

SITE FILE COPY

	Reviewed by:	Reviewed by:	Reviewed by:
Name	Sergio Lopez	Mike Edelman	George Purpura
Date	7-7-95	7-7-95	7-7-95
Signature		Dr. Mike Edelman	FAXED

SO123-606-1-9-0

MGP INSTRUMENTS

Suite 150
5000 Highlands Parkway
Smyrna, GA 30082

SO123-606-1-9-0

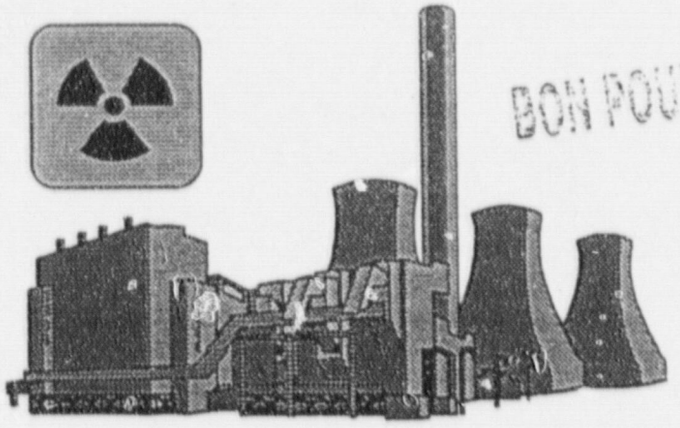
REVISION LOG:

Revision #	Date	Revised Pages	Comments
0	7/6/95	N/A	Original Issue

TABLE OF CONTENTS

MGP INSTRUMENTS

Radiation Monitoring System



RMS Protocol Technical Specifications

Written by:	J. Nadaud	Date:	16/02/94	Visa:	
Checked by:	A. Pommier	Date:	16/02/94	Visa:	
Approved by:	L. Meslay	Date:	16/02/94	Visa:	

INTERNAL DISPATCHING				EXTERNAL DISPATCHING
F. Chlosta		J. Nadaud		MGP Instruments Inc.
F. Schulcz		G. Ruaro		
S. Dimanche		A. Pommier-		
P. Stolpe				

Number of pages : 28

B	06/06/95	Update		J. Nadaud	S. Dimanche	L. Chapelot
A	16/02/94	Original edition	MF	Writer	Checker	Approb.
Ind.	*Date*	*Modification number & designation*	*M.F.*	*Signatures*		

45866 BA

A	16/02/94	Original edition		Writer	Checker	Approb.
Ind.	*Date*	*Modification number & designation*	*M.F.*	*Signatures*		

Update Table

Index /Date Written by	Modified pages	Origin and designation of the modification
B - 06/06/95 - NAJ	All	Merlin Gerin Provence changed to MG Instruments
	p15	Exception code clarification
	p23	The protocol accepts 2 stop bits
	p12	No response time-out is speed dependant
	p27	Changed crossed cable to straight cable

Table of content

1. Introduction

1.1. Purpose

All units of the Radiation Monitoring System integrate digital communication links to be able to communicate between each other. Each unit has its own data stored in a parameter table that is shared between the unit itself and an external device (as a master unit, or a computer).

To allow external devices to read and/or write into this parameter table, each unit integrates a communication protocol compatible with MODBUS/JBUS.

The purpose of this document is to describe this protocol as used in RMS. A unit that wants to be connected to the Merlin Gerin Radiation Monitoring System, using embedded protocol, shall comply with these specifications.

Additional documents will be written for optional communication protocols that may be implemented on extension boards.

1.2. Scope

This technical specification concerns the whole Radiation Monitoring System project since each unit manages an embedded protocol compatible with MODBUS.

1.3. Definitions, Acronyms and Abbreviations

ANSI	*American National Standards Institute*
CM	*Measurement board*
CRC	*Cyclic Redundancy Code*
IEEE	*Institute of Electrical and Electronics Engineers*
LAN	*Local Area Network*
LDU	*Local Display Unit*
LPU	*Local Processing Unit*
LSB	*Least Significant bit*
MASS	*Maintenance And Setup Software*
MSB	*Most Significant Bit*
NU	*Not Used*
PC	*Personal Computer (IBM compatible)*
RDU	*Remote Display Unit*
RMS	*Radiation Monitoring System*
SCADA	*Scan, Control, Alarm and Data Acquisition computers*

1.4. References

[1]	**ANSI/IEEE Std 729-1983**	*IEEE Standard Glossary of Software Engineering Terminology.*
[2]	**MGP-SQAP - 45203**	*RMS Software Quality Assurance Plan*
[3]	**MGP-SDP - 45202**	*RMS Software Development Plan*
[4]	**MGP-GRS - 45254**	*RMS General Requirements Specification*
[5]	**MGP-SSRS - 45179**	*System Software Requirements Specification*
[6]	**MGP-LPU-SRS - 45180**	*Common LPU Software Requirements Specification*
[7]	**MGP-DU-SRS - 45182**	*DU Software Requirements Specification*

1.5. Overview

This document describes the embedded communication interface for the RMS project. It does not concern communication board options.

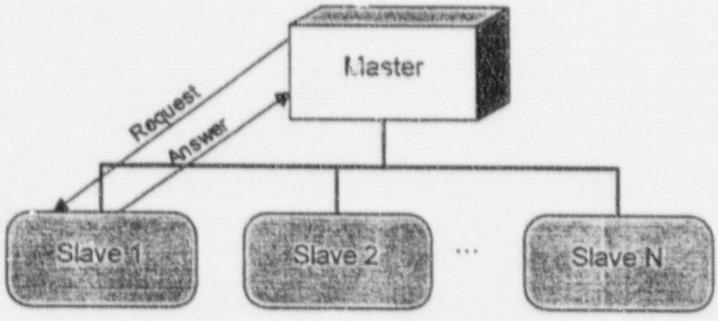
We will first present the protocol principle, then describe implemented function in detail, and give information about communication lines supported by this protocol.

2. Protocol definition

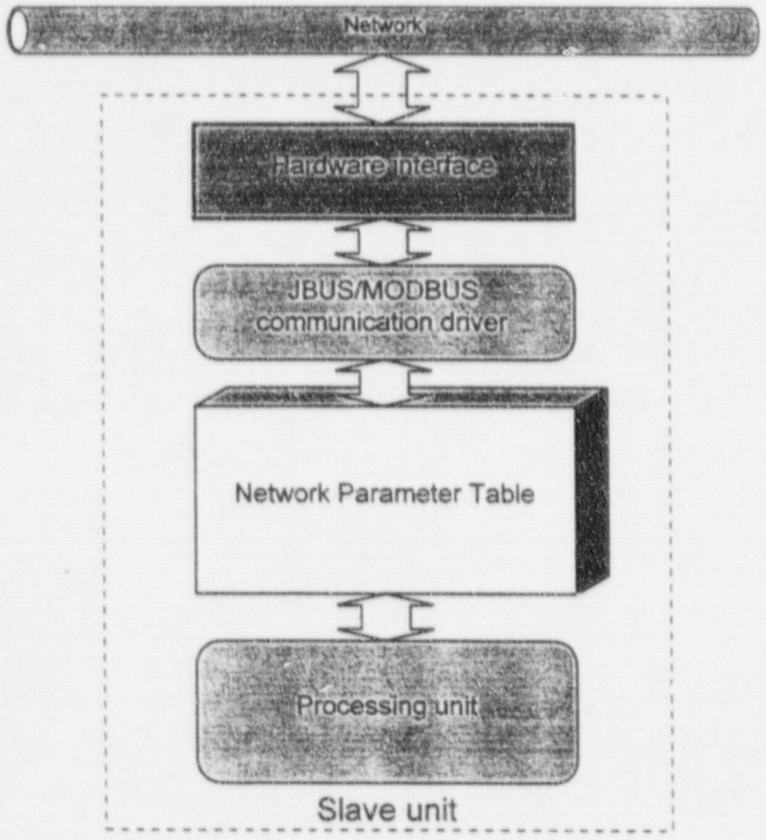
2.1. Introduction

All the RMS devices have several digital communication links. The communication protocol is MODBUS/JBUS.
The protocol has to be compatible with JBUS (as defined by APRIL) and with MODBUS (as defined by MODICON).

MODBUS protocol allows communications between a master station and up to 255 slave stations, using point-to-point or multidrop connections. There is only one station, the master, that is allowed to start an exchange. Each station must have a specific address on the network (slave identification).



Each slave unit has an internal memory, called network parameter table, that is shared by the processing unit, and by the network driver.

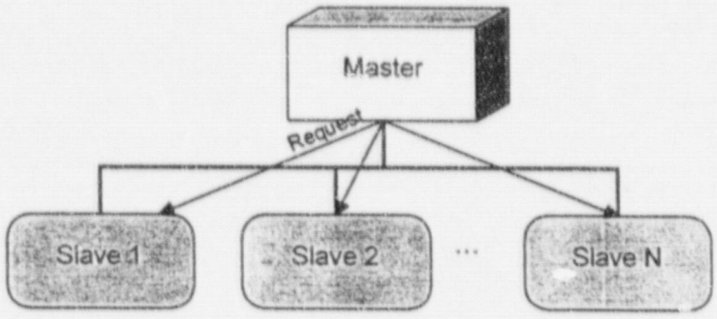


This parameter table is accessible through the network by using read and write functions. Because MODBUS protocol manipulates 16 bit words with 16 bit addressing, that allows the table to handle up to 65 536 words (or 128 K bytes).

The protocol includes the following communication functions:

Function number	*Description*
3,4	Read N words from the slave memory.
6	Write 1 word in the slave memory.
16	Write N words in the slave memory.
8	Communication diagnosis function (code 10 to 18).
11	Event counter reading.

It accepts the writing diffusion capability: Possibility to write the same data to all the slaves at one time by using the address 0. There is, in this case, no response from slaves.

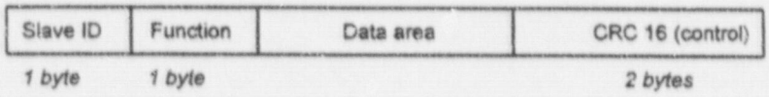


2.2. Exchange characterization

An exchange is composed of two messages:

- A request from the master
- A response from the slave

Each message has the same frame (4 types of data):



1. **Slave ID (1 byte)**

   This slave number identifies the destination (1 to 255). If the number is zero, the request concerns all slaves, there is no response.

2. **Function code (1 byte)**

   To select the command (reading, writing...) and to check if response is correct.

3. **Data area (n bytes)**

   This information area contains function related parameters: word address, word value, number of words.

4. **Control word (2 bytes)**

   Used to detect transmission errors. See annex 1 for CRC16 calculation.

☞ ***Note***: The frame length is limited to 255 bytes maximum. This is for timing constraint, data integrity assurance and buffer sizing. If length of the received frame exceeds 255 bytes, no response is sent.

2.3. Timing characterization

2.3.1. Exchange synchronization:

- If a character is received after a silence of more than 3 characters, it is considered as a frame start.
- There must be a silence on the line of minimum 3 characters between two frames.

Each unit when receiving a message has to:

1. Read the query message.
2. Ensure that the request is addressed to the unit.
3. Check the message integrity (CRC 16).
4. Check if the request is valid (good function number...).
5. Execute the action associated with the request.
6. Elaborate an answer.
7. Send the answer.

Everything has to be done within 30 ms to minimize the response time.

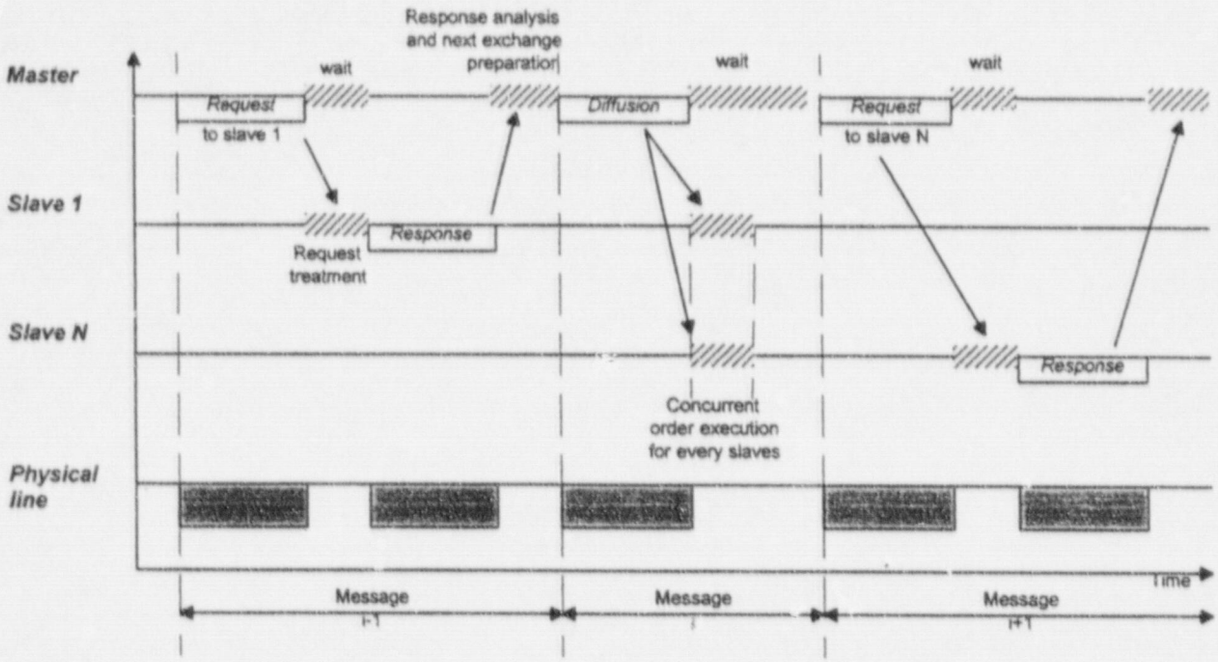
To detect the end of frame, the unit measures the time between every character we receive: If there is a silence of more than 3 characters, we consider that the message is ready. Here is the table of end of frame time-out according to the transmission speed.

Speed (BPS)	9600	19200	28800	38400	57600	115200
Time-out (ms)	4	3	3	3	3	3

For the diffusion command, we have to wait 50 ms before sending another message. This delay allows the slave unit to end its frame treatment without being disturbing by a new frame.

☞ ***Note***. The minimum speed on the network is 9600 baud (for minimum communication efficiency), the speed by default is 19200 (acceptable on every unit or computer). The maximum speed on the line, 115 kBauds (RS485 specification) is not acceptable on every unit because it depends on the capability the unit has to process communication frames. On RMS, LDU and DU units are limited to 57600 baud.

2.3.2. Network load analysis



Note: Request, response, diffusion, waiting and treatment duration depends on realized function.

2.3.3. Maximum exchange duration

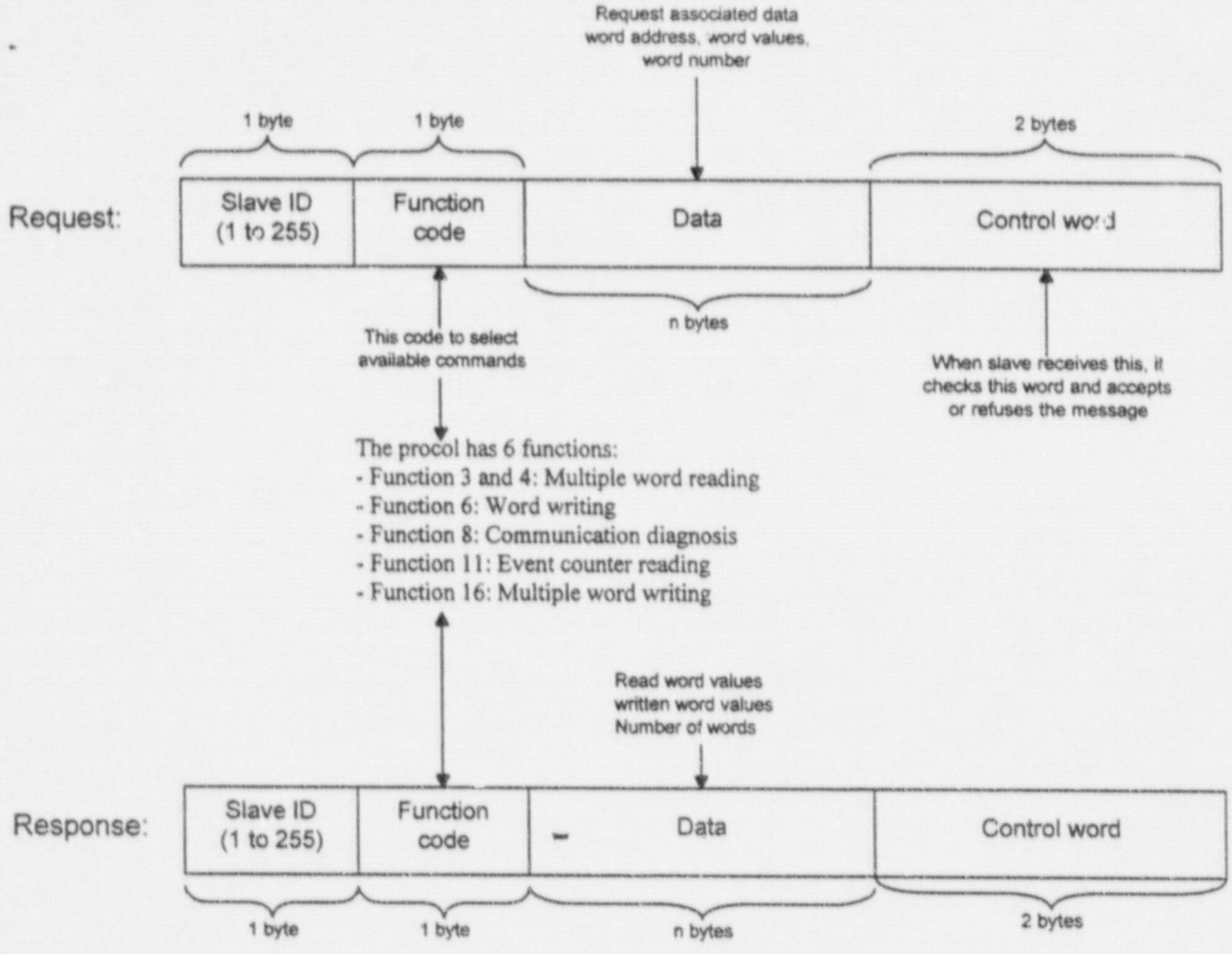
If previous conditions are respected, here is the slowest exchange: write 123 words at 9600 baud.

Request frame length = 9+246=255
Response frame length = 8

At 9600 baud we consider that it takes 1ms to transmit one character:
Exchange duration =
255 (request) + 30 (treatment) + 8 (response) + 30 (treatment) ≈ 330 ms

The command time-out, maximum duration between the end of request transmission, and the beginning of the response reception, has to be in all case higher than the maximum exchange duration.

2.4. Request and response frame presentation



2.5. Slave received message control

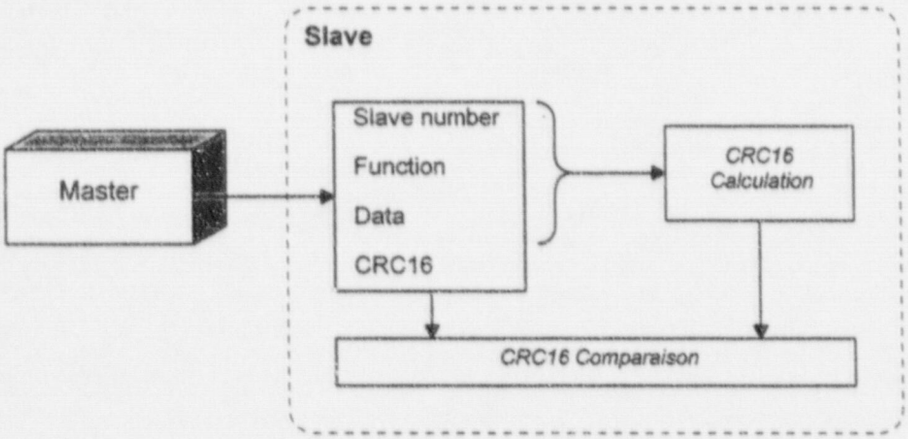
2.5.1 Message control process

When the master sends a request with following information:

- slave number (identification),
- function code,
- function parameters.

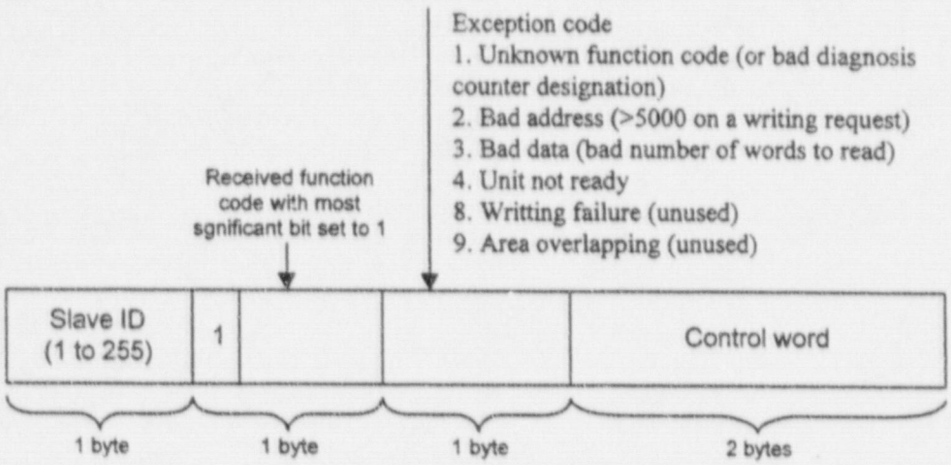
It calculates and sends the control word (CRC16).

When a slave receives the request message, it stores it in memory, calculates the CRC16 and compares it to the received control word.



If received message is not correct (CRC16s are not equal) the slave does not respond.
If received message is correct, but the slave cannot treat it (bad address, data...), it sends back an exception message.

2.5.2. Exception message content



Example: Try to send an unknown function (# 9)

Request

1	9	0	0	0	0	CRC16

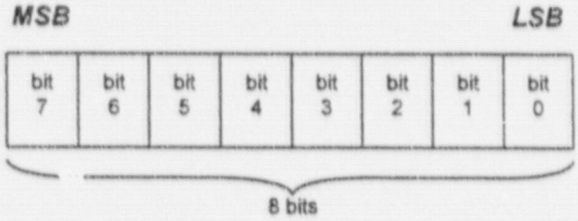
Response

1	89h	1	CRC16

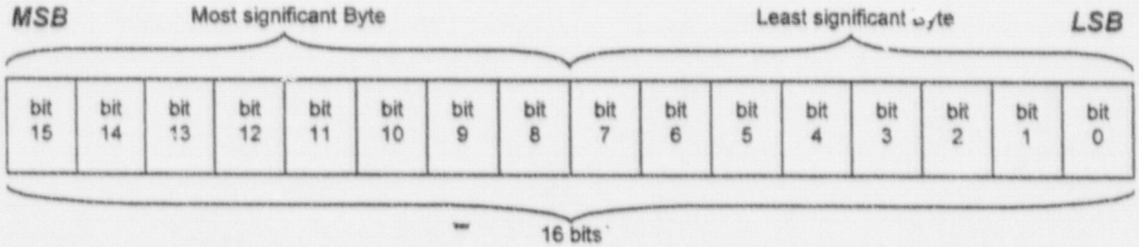
2.6. Data representation

The protocol manipulates 8 bit characters (coding from 0 to 255) on the line, but information that is to be sent or received is 16 bit word size.

2.6.1. Byte (or character) representation



2.6.2. Word representation



The most significant byte is sent first on the line, then the least significant byte.

3. Protocol functions

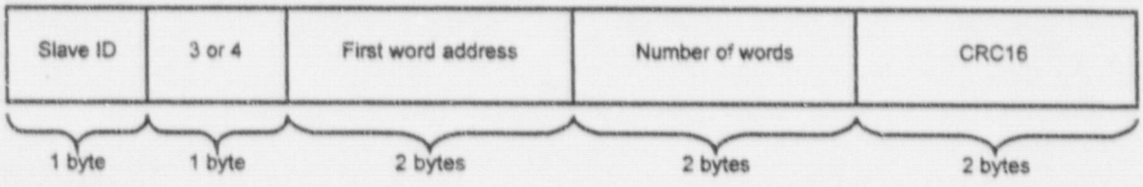
3.1. Multiple word reading: function 3 or 4

The number of words to read must be lower or equal to 125.

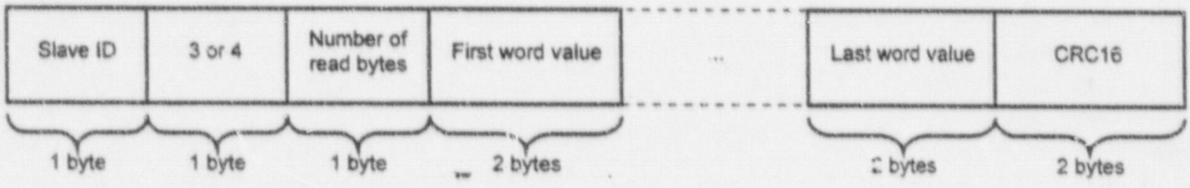
- Function 3: Read internal words.
- Function 4: Read external input words.

In fact, these two functions are completely similar. It is then possible to use either one without any change. There are both implemented for MODBUS/JBUS compatibility purpose.

Request:

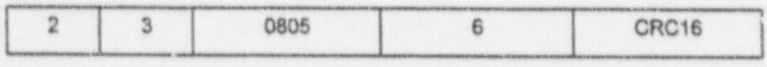


Response:

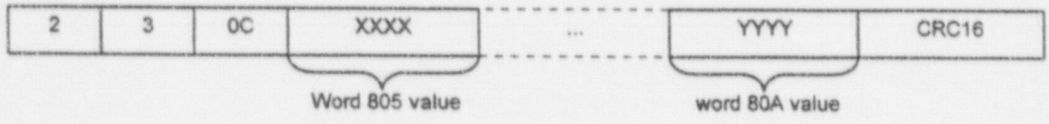


Example: Read words 805 to 80A from slave #2

Request



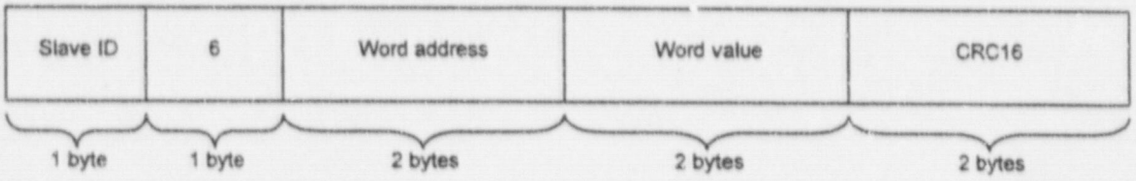
Response



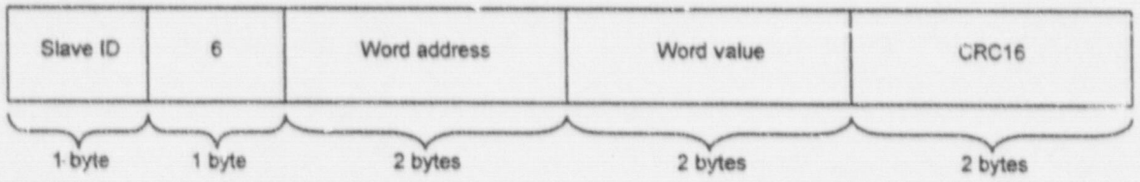
3.2. Word writing: function 6

This is the faster way to write one word into slave memory. To write two or more adjacent words, it is better to use the function 16.

Request:



Response:



Response is the request echo signaling that the word has been correctly written into slave memory.

Note: If slave ID is 00, all slaves execute the command without sending back any response.

Example: Write 1000 at 810 in slave #1

Request

1	6	810	1000	CRC16

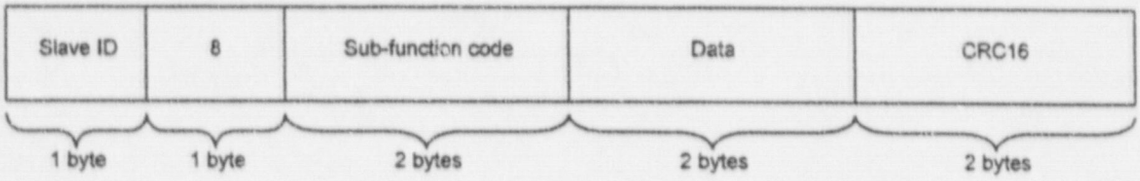
Response

1	6	810	1000	CRC16

3.3. Diagnosis counter reading: function 8

Each slave manages some diagnosis or event counters. There are nine counters per slave. Each counter is a 16 bit word. These counters are updated when a frame is received and processed by the unit (the message is addressed to the unit) and if this is not counter reading requests (function 8 and 11).

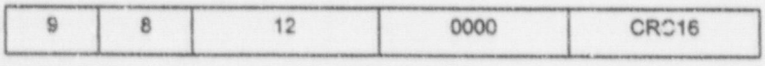
Request / Response:



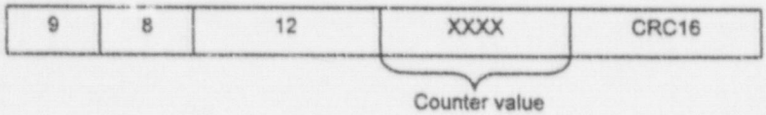
Command	Sub-function code	Data	
Diagnosis counter reset	10	0000	
Read total number of:			
• Received frames without CRC error	11	XXXX	
• Received frames with CRC error	12	XXXX	On request frame,
• Exception responses	13	XXXX	XXXX is set to 0000,
• Frames addressed to the unit (except broadcast)	14	XXXX	on response, XXXX contains specified
• Received broadcasted requests	15	XXXX	counter value
• Not used but implemented for compatibility (always set to 0000)	16	0000	
• Unit not ready responses (unused)	17	0000	
• Character not processed (unused)	18	0000	

Example: Read number of frames received with CRC error from slave #9

Request



Response



3.4. Event counter reading: function 11

Each unit (slave or master) has an event counter that is incremented each time it receives and processes correctly a message (except for specific functions that read diagnosis counters: function 8 and 11).

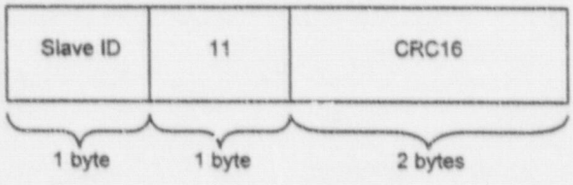
A diffusion command also makes the counter running. If a slave sends an exception command, counter is not incremented.

This counter allows, from the master, to know if the slave correctly processed the command (event counter incremented), or if the slave did not interpret the command (event counter unchanged).

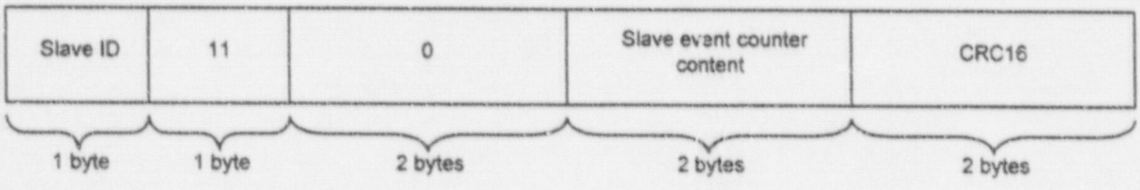
By reading these different variables, it is possible to get a protocol diagnosis between master and slave.

- If master counter equal slave counter, command sent by the master has been correctly processed.
- If master counter equal slave counter plus one, command sent by the master has not been executed.

Request:



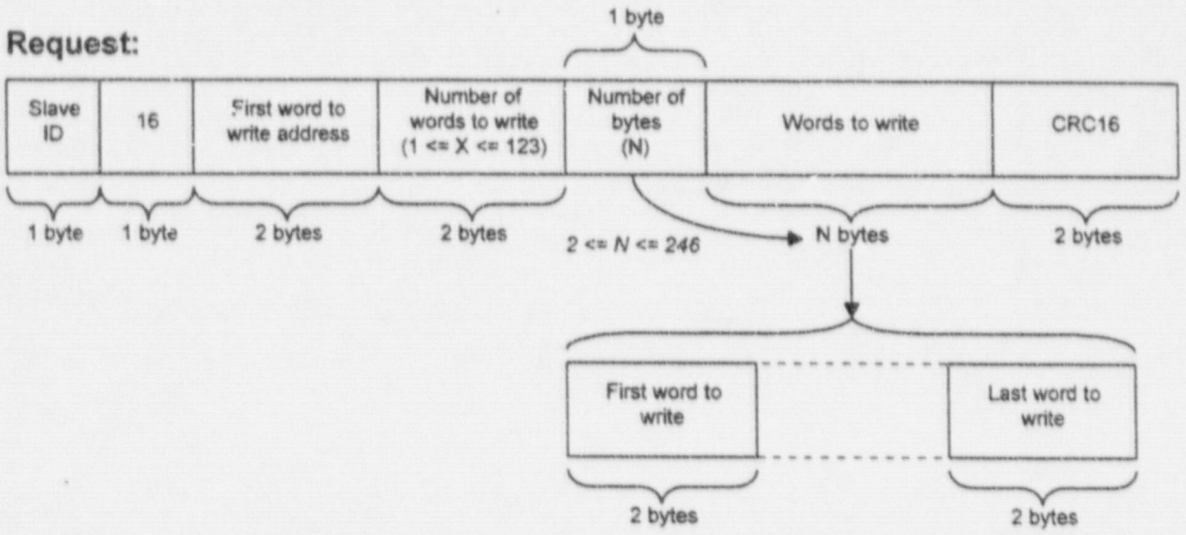
Response:



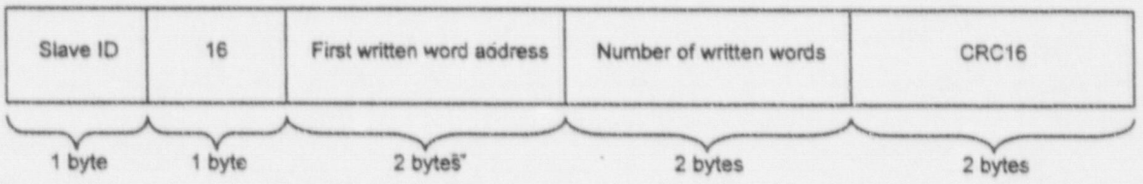
3.5. Multiple word writing: function 16

This function is useful to write complete data structures to slave memories. It is possible to write up to 123 words (246 bytes) at once.

Request:



Response:



☞ ***Note***: If slave ID is 00, all slaves execute the command without sending back any response.

Example: Write following data at 800 in slave #1

(800) = 10
(801) = 100
(802) = 1000
(803) = 10000

Request

1	16	800	4	8	10	100	1000	10000	CRC16

Response

1	16	800	4	CRC16

3. Communication settings

The transmission speed by default is 38400 BPS.

The character format is:

1 start bit
8 data bit
No parity
1 or 2 stop bit

The transmission speed is setup by software and must include the following speeds: 9600,19200, 28800, 38400 and 57600 BPS.

This protocol is half-duplex, and does not need any hardware or software control (CTR/RTS, XON/XOFF). This is a binary protocol that uses the full 8 bit character size (256 different possible states for one character). This is therefore not possible to communicate in ASCII mode.

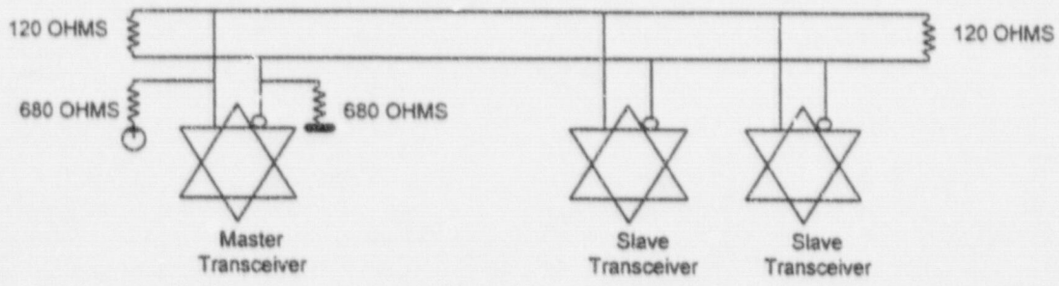
4. Hardware interface

The physical interface has to conform with the RS485 or RS232C norm.

4.1. RS485 Multidrop interface

The resident communication interface meets the RS-485 standard.

The RS-485 standard accommodates the requirements on a balanced transmission line used in party-line circuit configuration. It permits Multidrop applications where multiple drivers and receivers share the same line in data transmission.
The transmission line which is intended to be 120Ω twisted pair is terminated at both ends. The drivers and receivers can be distributed between the termination resistors.

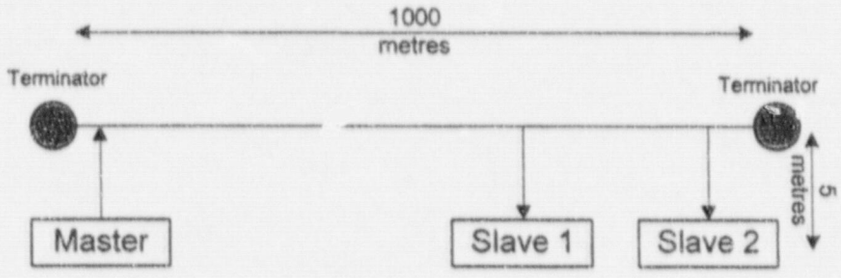


The communication will operate properly in the presence of reasonable ground drops, withstand line contention situations and carry 32 drivers and receivers on the line.

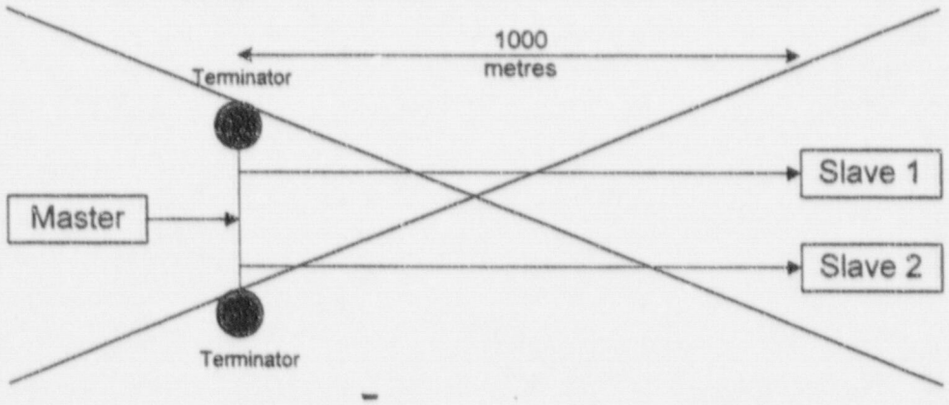
Because this is a bus type network, there are some topology constraints to respect.

Here are some good and bad configurations:

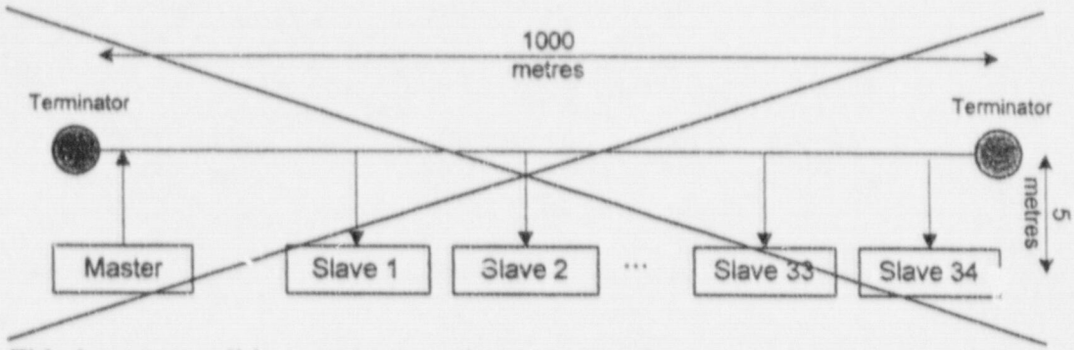
1- Bus topology with right distances (good configuration)



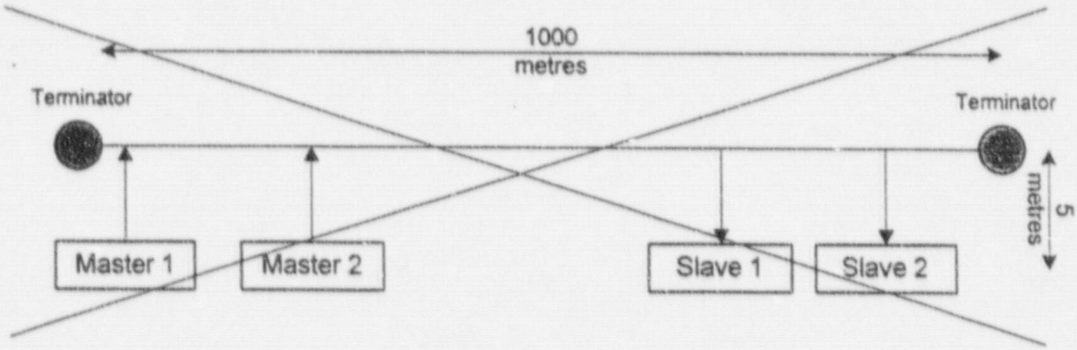
2- Distances are not respected, this bad configuration looks like a star topology.



3- Bus topology with right distances, but too many slaves (maximum of 32 stations on the same wire)

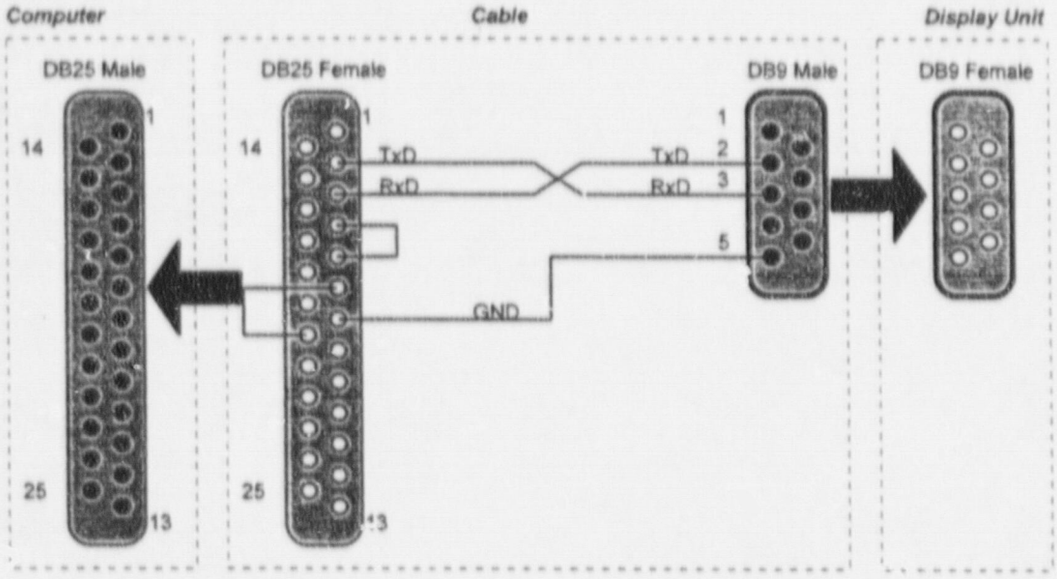


4- This is not possible to put more than one master on the same network



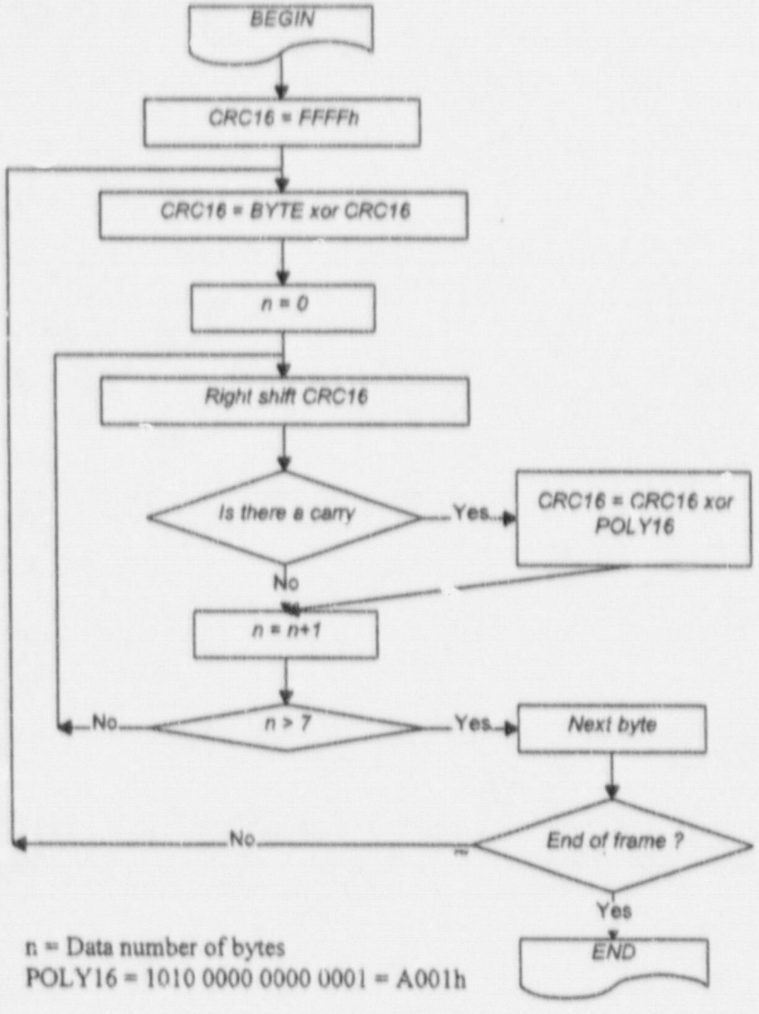
4.2. RS232 point-to-point interface

Serial Cable to connect a PC with a DU (25 to 9)



DB9 pins	EIA designation	Description	DIN Designation	DB25 pins
1	DCD	Data Carrier Detect	M5	8
2	RD	Transmitted Data	D1	3
3	TD	Received Data	D2	2
4	DTR	Data Terminal Ready	S1.2	20
5	GND	Signal Ground	E2	7
6	DSR	Data set Ready	M1	6
7	RTS	Request To Send	S2	4
8	CTS	Clear To Send	M2	5
9	RI	Ring Indicator	M3	22

Appendix A: CRC16 algorithm



Note: For CRC16 word, the first byte that is sent is the least significant one.

Example: Write 810h at 1000h in slave 1

Request frame

01h	06h	08h	10h	10h	00h	87h	AFh
header		Address		Value		CRC16	