TOTAL ENGINEERING SERVICES TEAM, INC.
(TEST Inc.)

A Weatherford Enterra Company

SCADAWARE™

ALLEN-BRADLEY PLC PROTOCOL INTERFACE

Document 1250-02

Revised: April 1996

This document is (C) Copyright 1994 by
Total Engineering Services Team, Incorporated,
(TEST Inc.), New Orleans, La. USA
All Rights Reserved

Contact Arthur Zatarain, PE
via www.artzat.com
For information on this document

CONTENTS

INTRODUCTION

Allen-Bradley PLCs can receive command messages and reply to them via Data Highway, Data Highway Plus, DH-485, or RS-232 network. By using the appropriate interface module, a TEST SCADA system can access the network and communicate with any PLC using the Allen-Bradley protocol.

This document provides detailed information on TEST's implementation of the Allen-Bradley communications protocol. Although the user is not expected to be an expert, some familiarity with Allen-Bradley PLCs and TEST SCADA systems is assumed. This document is not a tutorial for Allen-Bradley PLCs. For additional information about A-B PLCs and interface modules refer to the following A-B publications:

Publication 1770-6.5.16	Data Highway/Data Highway Plus/DH-485 Communication Protocol and Command Set Reference Manual
Publication ICCG-11.6	Allen-Bradley Communication Division SCADAApplications Guide
Publication 1770-6.5.13	Data Highway or Data Highway Plus Asynchronous (RS-232-C or RS-422-A) Interface Module (Cat. No. 1770-KF2) User's Manual
Publication 1785-6.5.2	Allen-Bradley 1785-KE Data Highway Plus Communications Interface Module User's Manual

PLC-5 HARDWARE INTERFACE

The TEST SCADA software uses a standard RS-232 port for communication. This port can be connected to an Allen-Bradley PLC-5 in one of the following three ways:

1 A 1784-CP10 or 1784-CP11 interconnect cable can be used for a direct connection to Channel 0 on a PLC-5. This provides communication to a single PLC on a peer to peer basis. Each command received by the PLC through channel 0 is processed without regard for the Destination ID included in the command. The Source ID included in each command is actually used to specify a file number in the PLC's memory for which the command applies.

2 A 1770-KF2 module can be used to connect to Channel 1A on a PLC-5 and communicate over the Data Highway Network. When using this module, the Source ID included in the command is replaced with the address of the 1770-KF2 module before being sent to the intended PLC. Since the Source ID is actually used to specify a file number in a PLCs memory, only a single file can be accessed for each PLC connected to the 1770-KF2 module. This file is called a "Compatibility File" and its number is the same as the address of the KF2 module. The bit modify and move instructions BTD, MOV and MVM can be used in a PLC's ladder logic to transfer bits to or from the compatibility file to other areas of a PLC's memory.

3 A 1785-KE module can be used to connect to Channel 1A on a PLC-5 and communicate over the Data Highway Network. This module contains a dip switch which can be set for either LOCAL or REMOTE mode. When in Local mode the 1785-KE module functions just like the 1770-KF2 module mentioned above. However, when in Remote mode the 1785-KE module will not replace the Source ID with its own address before sending the command to the intended PLC. It will simply pass along whatever Source ID is included in the command just as if a 1784 interconnect cable were being used.

FULL vs HALF DUPLEX

Full duplex is used for point to point links where two devices communicate on a peer to peer basis. This involves simultaneous two-way communications and both stations on the link can initiate communications whenever they wish. Full Duplex can also be used to communicate to a number of remote stations on the Data Highway using point to point links.

Half duplex is used for multidrop networks where there is one master station and one or more slave stations. All stations are tied together on a common link, and thus everybody hears everything transmitted on the link. Since there is potential for two station's transmissions to collide if they both try to transmit at once, only the master can initiate a message any time it wishes. The slaves can only communicate when polled by the master.

For communications between a number of stations, either Full duplex or Half duplex could be used. Using Full duplex will provide the best performance. However, it is more expensive since communications to the remotes are via dedicated point to point links. Half duplex is based on full-duplex protocol but also includes a Station ID at the beginning of each master message.

When the TEST SCADA software is acting as a master using half duplex protocol, the Station ID included in each outgoing message is provided by the MAP SCAN or MAP DATA command. The station ID represents the address of a slave on a multidrop link. This should not be confused with the Destination ID which is the final destination of the message.

In most situations, a PLC is actually a slave on the multidrop link and the final destination of a message. Therefore, the SRC and DST are the same. However, when multiple PLCs are tied into a network interface module, the interface module acts as a slave on the multidrop link and the PLCs are the ultimate destinations of the messages. In this situation, the MAP SCAN and MAP DATA commands should specify the Station ID to be the address of the interface module and the Destination ID to be the address of the intended PLC. Refer to the MAP SCAN command for more information on how to specify a Station ID.

MAP SETUP FILE KEYWORDS

The map file contains plain text statements on lines which begin with Map File keywords. Each line performs a specific function to build and define the tables which form the data map between TEST SCADA data types and Allen-Bradley PLC data types. The map file can be processed by any task, although normal practice is to have it processed by Task 0 during system startup.

The format of the file is very similar to other TSP test files. The keyword is the first word on the line, and some abbreviations are allowed. A semicolon (;) can be used as a comment character, allowing for comments anywhere in the text file.

MSG Text message

Text messages can be sent to the system console during map file processing with the MSG statement. All text after the MSG keyword is echoed to the system console.

TAG Table Reference ID

Set the tag name for the data map being defined. By default, the tag names for each data map are MAP1, MAP2, MAP3, and so on. These tag names can be used by any RTU type task to select a data map by tag name rather than by index number.

```
Tag PLC1          ; Tag for this table is PLC1
```

ERROR Specify BCC or CRC Error Checking

Specify the type of error checking that will be used to check the accuracy of each message packet transmission. The two types of error checking available are the block check character (BCC) and the 16-bit cyclic redundancy check (CRC-16). The keywords BCC and CRC are used to specify the type of error checking to use. The default is BCC.

```
Error CRC         ; Specify CRC error checking
```

ACK Control Message Acknowledgements

Specify ON, OFF, or AUTO to control message acknowledgements to a PLC. Specifying ON will cause an ACK to be sent to a PLC after each message is successfully received. Specifying AUTO will cause an ACK to be sent to a PLC before each message is sent and after each message is successfully received. The default setting is OFF which prohibits any ACKs from being sent and can significantly reduce the number of necessary transmissions.

```
ACK ON            ; send ACK after each message is successfully received
ACK AUTO          ; send ACK before sending msg and after receiving msg
```

DUPLEX Specify Full or Half Duplex Protocol

Specify whether FULL or HALF duplex protocol will be used. The default setting is Full Duplex. For more information refer to section Full vs Half Duplex.

```
DUPLEX HALF       ; use Half duplex protocol
```

STATION Specify Station ID

Specify the Station ID for the current map table. This ID is only necessary when using half duplex protocol. In half duplex, the beginning of each message sent by a master contains a Station ID. When this unit is acting as a slave and a message is received, a check is done to see that the Station

ID included in the message matches the Station ID of this unit. If it does not match the message is ignored.

NOTE: This Station ID is only used to check incoming messages, it is not included in any response to incoming messages or in outgoing messages initiated by this unit acting as a master. When acting as a master, the Station ID included in any outgoing message is provided by the MAP SCAN or MAP DATA command itself.

```
STATION 8                    ; specify STN address of this map
```

UNITS Multi PLC Emulation

Within each data map, up to 16 PLC IDs can be defined, each with its own map setup. The UNITS command defines how many logical Allen-Bradley PLCs will be emulated by the data map being defined. If this command is omitted, a value of 0 is assumed.

```
Units 5          ; allow 5 different PLC addresses to be emulated
```

ID Set PLC Address

Specify the PLC ID number to be used for subsequent data maps as well as the maximum number of files that can be mapped for that PLC. No table sizes or data maps can be defined until an ID has been specified. Valid IDs range from 0 to 254 and each ID command will begin the next map setup. Data can be mapped for any PLC file starting with 0 and ranging up to the maximum number of files specified minus one.

```
ID 4 9    ; start defining PLC address 4 with a maximum of 9 files (0-8)
```

SELE Select Logical TSP RTU

Select a logical TSP RTU which is used in subsequent TSP data table references. This is a convenience so that subsequent map lines do not have to explicitly name a particular RTU on each line. The RTU named in the SELECT line will be the default RTU.

```
Sele WD34A        ; map following chans for West Delta 34 A RTU
```

FILE Define a PLC Map File

Define a PLC file which is used to map specific PLC data types with TSP data types. Parameters include a file number, a keyword specifying the type of file and a number specifying the number of elements in the file. Available file numbers range from 0 to the maximum number of files specified in the ID statement minus one. There are only 4 PLC data types supported at this time and the available keywords are:

```
OUTPUT       - digital, off or on
INPUT        - digital, off or on
INTEGER      - 16 bit integer
FLOAT - 32 bit floating point
```

If more than one file definition exists for the same file number under the same map ID, only the first definition line for that file will be accepted.

```
FILE 0 OUTPUT 256   ; define file 0 as type Output with 256 elements
FILE 1 INPUT 256              ; define file 1 as type Input with 256 elements
FILE 7 INTEGER 128  ; define file 7 as type Integer with 128 elements
FILE 8 FLOAT 64               ; define file 8 as type Float with 64 elements
```

MAP STATEMENT Define map between TSP and PLC data

Specify a link between TSP channel data and PLC file data. Parameters include a PLC file number, a number representing the position within that file, a TSP channel or channel range and optional data type specification. The specified PLC file number must be defined using the FILE statement before it can be mapped.

Following the PLC file number is a number which represents the position within that file where the TSP channel mapping will begin. Remember, data within a PLC file starts at position 0 and extends up to the maximum number of elements in the file minus 1.

The next parameter can be either a single TSP channel or a channel range. If a single channel is given it is directly mapped with the specified PLC file and position. If a channel range is specified, the first channel in the range is mapped to the specified PLC file and position. Consecutive channels in the channel range are mapped to consecutive positions in the same PLC file.

For example, consider the following lines which could be used to map TSP Output channels 1-5 with PLC Output file 0 positions 0-4. In this example each line maps a single TSP Output channel.

```
MAP 0 0 01          ; Map PLC file 0 position 0 with TSP channel 01
MAP 0 1 02          ; Map PLC file 0 position 1 with TSP channel 02
MAP 0 2 03
MAP 0 3 04
MAP 0 4 05
```

By using a channel range, the same 5 TSP Output channels could be mapped to the same 5 Output file positions in a single command as shown below:

```
MAP 0 0 01:05    ; map chans 01-05 with file 0 positions 0-4
```

The remaining parameters of a map specification are strictly optional. These parameters are keywords which specify exactly what data pertaining to the TSP channels will be mapped to the PLC files.

MAP STATEMENT OPTIONS

For PLC **Output** and **Input** files, each file location contains either a 0 or a 1 Therefore, each location can be mapped with the value of a digital TSP channel (Status Input or Output) or the status of a channel condition, such as the existence of a particular alarm condition. The available keywords for specifying which channel data will be mapped into each PLC Output and Input file location are listed below. Following each keyword is the condition for which the corresponding PLC locations will contain a 1 when the data is transferred from TSP.

VALUE	- if current value of channel < > 0
NEW	- if in alarm but not yet acknowledged
ALARM	- if acknowledged but still in alarm state
DB	- if cleared but still within deadband
RES	- if no longer in alarm and waiting for reset
ABNORMAL	- if alarm conditions NEW, ALARM, or RESET exist

If no keyword is specified the keyword VALUE is assumed. For PLC Outputs and Inputs, multiple keywords can be specified on a single line. In such a case, the associated PLC file location would contain a 1 if any of the conditions are met at the time of the transfer from TSP to the PLC. For example, consider the following 3 map setup lines where file 0 is a PLC Output file:

MAP 0 0 01 Value
MAP 0 1 01 New
MAP 0 2 01 Value New

Now consider the value of PLC Output locations 0, 1, and 2 after an update of these 3 locations from a TEST SCADA unit. If the value of Output channel 1 is 1, PLC Output file 0 position 0 would contain a 1, otherwise a 0. If Output channel 1 is in alarm but has not yet been acknowledged, PLC

Output file 0 postion 1 would contain a 1, otherwise a 0. If either the value of TSP Output channel is 1 or Output channel 1 is in alarm but has not yet been acknowledged, PLC Output file 0 postion 2 would contain a 1, otherwise a 0.

An update from a TEST SCADA unit to a PLC can occur in 1 of 2 ways. First, a request can be sent to a TEST unit acting as a slave. The TEST unit would interpret the request, build up a response in the PLC format, and send back the response. Second, the TEST SCADA unit can act as a master and simply build up a PLC command and send it out.

Now let us consider what happens when data is transferred in the opposite direction, from a PLC to a TEST SCADA unit. When data is transferred in this direction, only the PLC locations that are mapped to TEST SCADA channel VALUES will have an effect on the TEST SCADA unit. For example, again consider the following 3 map setup lines where file 0 is a PLC Output file:

MAP 0 0 O1 Value
MAP 0 1 O1 New
MAP 0 2 O1 Value New

Now consider the state of the TEST SCADA unit after an update from these 3 PLC locations. If the value of PLC Output file 0 position 0 is 1, Output channel 1 1 would contain a 1, otherwise a 0. The update from PLC Output file 0 position 1 would have no effect on the TEST SCADA unit because the mapping does not contain a channel value. The update from PLC Output file 0 position 2 would have the same effect as the update from PLC Output File 0 position 0. This is because the NEW parameter would be ignored and the VALUE parameter would work the same as it did for position 0.

There is one last point worth mentioning about the mapping of PLC Output and Input files. Since these locations contain either a 0 or a 1, they are easily mapped with the Status Input and Output channels of a TEST SCADA system. However, PLC Output and Input files can also be mapped to any other type of TEST SCADA channel. Transferring data for value type channels is exactly the same as transferring data for digital type channels. When going from a TEST SCADA unit to a PLC, the PLC location will be set to 1 when the value of the corresponding channel is not equal to 0, otherwise it will be set to 0. When going from a PLC to a TEST SCADA unit, a channel value will be set to 1 when the PLC location contains a 1, otherwise it will be set to 0.

For PLC **Integer** files each location contains a 16 bit integer value. What each 16 bit value represents depends on the map setup for each location. The available keywords for specifying what data will be mapped into each PLC Input file location are listed below.

VALUE - integer representation of channel's current value
HI - hi alarm setpoint
LO - lo alarm setpoint
DB - deadband value

If no keyword is specified the keyword VALUE is assumed. If more than one keyword is specified only the first one will have affect. Consider the following MAP setup lines where file 7 is a PLC Integer file:

MAP 7 0 V1 <- map integer value of V1 with file 7 position 0
MAP 7 1 V1 HI <- map integer format of hi setpoint with pos 1
MAP 7 2 V1 LO <- map integer format of lo setpoint position 2
MAP 7 3 A1 DB <- map integer format deadband with position 3

For PLC **Floating Point** files each location contains a 32 bit floating point number. The data that can be mapped to these locations and the keywords used to setup the map are the same as that for Integer files. Now consider the following MAP setup lines where file 8 is a PLC Floating Point file:

MAP 8 0 V1 <- map float value of V1 with file 8 position 0
MAP 8 1 V1 HI <- map float format of hi setpoint with pos 1
MAP 8 2 V1 LO <- map float format of lo setpoint position 2
MAP 8 3 A1 DB <- map float format deadband with position 3

Again, if no keyword is specified the keyword VALUE is assumed.

SAMPLE PLC MAP SETUP FILE

```
; Allen-Bradley PLC Test File
msg Allen-Bradley PLC Map File Start

error CRC                        ; use 16 bit CRC error checking
ack OFF                                 ; do not send any Acks
duplex FULL                      ; use Full duplex protocol
units 3                                 ; total of 3 PLCs to be simulated

msg Starting Unit 1
ID 17   10                       ; first unit is PLC ID 17 with maximum 10 files
sele ss180                       ; TSP channel mapping will be for SS180 RTU

msg Defining PLC Files
file 0 output 200                       ; output bit file
file 1 input 32                         ; input bit file
file 7 integer 10                       ; integer file
file 8 float 16                         ; floating point file

msg Mapping Output File 0
map 0 0 o1                       ; PLC Output file 0 bit 0 is value of TSP 01
map 0 1 o1 NEW                          ; PLC Output file 0 bit 1 is new alarm status of
TSP 01
map 0 2 o1 ALARM                 ; PLC Output file 0 bit 2 is in alarm status of TSP 01
map 0 3 o1 DB                    ; PLC Output file 0 bit 3 is deadband status of TSP 01
map 0 4 o1 RESET                 ; PLC Output file 0 bit 4 is reset status of TSP 01
map 0 5 o1 ABNORMAL              ; PLC Output file 0 bit 5 is abnormal status of TSP 01
map 0 6 o1 NEW ALARM       ; PLC Output file 0 bit 6 is New or In alarm status of TSP 01
map 0 7 o1 NEW RESET             ; PLC Output file 0 bit 7 is new or reset status of TSP
01
map 0 8 o2                       ; PLC Output file 0 bit 8 is value of TSP 02
map 0 9 o2 NEW
map 0 10 o2 ALARM
map 0 11 o2 DB
map 0 12 o2 RESET
map 0 13 o2 ABNORMAL
map 0 14 s1:s16                         ; PLC Output file 0 bits 14-29 are TSP channels
S1-S16
map 0 30 v1:v5                          ; PLC Output file 0 bits 30-34 are TSP channels
V1-V5

msg Mapping Input File 1
map 1 0 s1:s16                          ; PLC Input file 1 bits 0-15 are TSP channels
S1-S16 map 1 20 o1:o16                         ; PLC Input file 1 bits 20-35 are TSP
channels O1-O16

msg Mapping Integer File 7
map 7 0 a1:a10                          ; PLC Integer file 7 integers 0-9 are TSP channels
A1-A10

msg Mapping Floating Point File 8
map 8 0 v1:v8
map 8 8 a1:a8

msg Starting Unit 2
ID  5
sele ss222                       ; TSP channel mapping will be for SS222 RTU

msg Defining PLC Files
file 1 input 100

msg Mapping Input File 1
map 1 0 s1:s16

msg Starting Unit 3
ID 9
sele hi170                       ; TSP channel mapping will be for HI170 RTU

msg Defining PLC Files
file 0 output 100
file 1 input 20
```

```
msg Mapping Output File 0
map 0 0 O1:08

msg Mapping Input File 1
map 1 0 s1:s16

msg Allen-Bradley PLC Map File Done
```

SETTING UP A PLC PROTOCOL TASK

Setting up a task to use the Allen-Bradley PLC protocol involves a simple 3 step process. These steps are:

1. Define the task
2. Load the map file
3. Select the data map

STEP 1. Any RTU type task, except task 0, can be used to communicate using the A-B PLC protocol. A single line in the main configuration (DAT) file is all that is needed to define an RTU type task. Examples of such lines are:

TASK RTU Com1
TASK RTU AB-PLC

The keyword TASK indicates a new task definition and the keyword RTU indicates the type of task. The third item on the line can be up to 16 characters long and is simply the task name.

STEP 2. In order for the TEST SCADA system to interface with an A-B PLC, a file must be processed which "maps" TEST SCADA data types into A-B PLC data types. The details of what is contained in the map file are described elsewhere in this document. For now, we assume that the map file has already been created and that it is correct.

The MAP LOAD command must be used to process the map file and setup the mapping scheme. This command can be processed by any RTU type task, including task 0. Normally this command is processed by task 0 during program startup. This can be either in the START0.RTU file or a subroutine called from this file. Again, the details of the MAP LOAD command can be found elsewhere in this document in a section called MAP COMMAND.

STEP 3. Once a mapping scheme has been loaded it is accessible by any RTU type task. However, the default protocol for all RTU type tasks is the TSP protocol. In order for a task to access a data map and use the Data Highway protocol the MAP SELECT command must be used. This command is usually processed from within the STARTx.RTU file which automatically gets processed when the task is started. Once a valid data map is selected for an RTU type task, the protocol for that task is determined by the type of map selected. For more details refer to the MAP SELECT command in the section called MAP COMMAND.

MAP COMMAND IN TSP LANGUAGE

This command is used to load a map file, select a particular data map for a task, send data to and receive data from another unit using the A-B protocol, and list the configuration of a data map to a file. The MAP command can be processed only by RTU type tasks. The format of this command is the word MAP followed by a keyword and some additional parameters. The available keywords and an explanation of each are given below.

MAP SELECT xx

Up to 10 data mapping schemes can be defined on a TEST SCADA system at one time. However, each task can have access to only one data map at a time. This command is used to select a particular data map for an RTU type task. A data map can be selected by index number or tag name.

Valid index numbers range from 1 to 10 and the default tag name of each map is the keyword "TAG" followed by the index number. This tag name can be changed from within the map file during a map load.

By default, the current data map index for each task is 0. This means that there is no default data mapping scheme for any task and the default protocol for all RTU type tasks is the TSP protocol. Once a valid data map is selected for an RTU type task, the protocol for that task is determined by the type of map selected (Modbus or A-B protocol).

The command MAP SELECT 0 can be used by an RTU type task to disassociate itself with any data map. This will cause the protocol used by that task to be returned to the TSP protocol. If a third parameter is not specified in the command, a message will be sent to the task processing the command to show the current map index for that task.

```
MAP SELECT 1                ; select data map 1 for current task
```

MAP LOAD xx AB [filename]

This command is used to process a map file which defines a data map. In this command a valid index number in the range 1 - 10 must be specified in place of the xx shown above. Specifying a map tag name instead of an index number will not work for this command. Because the data map to be defined is specified in this command by an index number, the task processing the command does not have to select this map, or any data map, prior to processing this command.

Following the map index number is the keyword AB which specifies that an Allen-Bradley map is being defined. The last parameter on the line is optional and is used to specify the name of the map file to be processed. If no file name is specified, the default file name RTU.AB is assumed. If a file name is specified but does not contain an extension, the default file extension AB is assumed.

If an attempt is made to process a MAP LOAD command using an index number for which a map has already been defined, the command will be ignored and the current data map for the specified index number will remain unchanged.

```
MAP LOAD 1 AB TEST          ; process map file TEST.AB to define map 1
```

MAP DUMP [filename]

This command is used to list the configuration of a data map to a file. This command can be processed by any RTU type task, but two conditions must be met prior to processing the command if it is to be successful. First, the task processing the command must have a valid map selected. Second, the currently selected map must be defined from a prior MAP LOAD command.

The last parameter on the line is optional and is used to specify the name of the file to contain the listing of the map configuration. If no file name is specified, the default file name AB.MAP is assumed. If a file name is specified but does not contain an extension, the default file extension MAP is assumed.

```
MAP DUMP    ; dump currently selected data map to file AB.MAP
```

MAP SCAN stn:dst file start count

This command is used to request data from a remote unit using the A-B protocol. This command can be processed by any RTU type task except task 0. Before processing this command, the task should have a valid A-B data map selected which has been defined using the MAP LOAD command.

The first parameter on the line following the SCAN keyword specifies a Destination ID and possibly a Station ID. The Destination ID represents the address of the PLC that is the ultimate destination of the message. This ID must be defined for the currently selected data map at the time this message is processed.

For all full duplex and most half duplex applications, the Destination ID is all that needs to be provided by this parameter. However, when using half duplex protocol, this parameter can also include a Station ID which is designated using a colon (:) and the format STN:DST. If the STN: is not specified, the Station ID included in the outgoing message will be the same as the Destination ID.

Following the PLC ID is the file number for which data will be read from the PLC. This file number must be defined for the specified PLC ID.

The last two parameters that must be specified in the SCAN command are the starting location and number of locations within the map for which data is requested. All mapping starts at location 0. For example, if a file of 200 elements is defined , the actual locations that exist are 0 - 199. If the start location specified is not within the valid range an error will result. If the number of locations specified extends beyond the upper limit, the count will automatically be reduced to eliminate the overrun.

When a TSP MAP SCAN command is processed an A-B Unprotected Read command is generated. This command can generally be executed by any PLC node and is used to read words of data from any area of the PLC data table memory. The number of words to read is automatically calculated based on the type of data contained in the specified file, the starting location in the file and the number of locations to read.

Full Duplex Mode

```
MAP SCAN 4 1 0 8      ; read 8 elements starting at location 0 from file 1 of PLC 4
MAP SCAN 2 7 6 3      ; read 3 elements starting at location 6 from file 7 of PLC 2
```

Half Duplex Mode

```
MAP SCAN 4 1 0 8      ; read 8 elements starting at location 0 from file 1 of PLC 4 Station 4
MAP SCAN 2 7 6 3      ; read 3 elements starting at location 6 from file 7 of PLC 2 Station 2
MAP SCAN 5:2 7 6 3    ; read 3 elements starting at location 6 from file 7 of PLC 2 Station 5
```

MAP DATA stn:dst file start count

This command is used to send data to a remote unit using the A-B protocol. The parameters required by this command are the same as those required by the MAP SCAN command with the exception that data will be written to the specified file rather than read.

When a TSP MAP DATA command is processed either an A-B Unprotected Bit Write or an Unprotected Write command is generated. The resulting command is determined by the type of file specified in the command. If the specified file is either an Output or Input file an Unprotected Bit Write command is generated. Otherwise, an Unprotected Write command is generated.

Full Duplex Mode

```
MAP DATA 4 1 0 8      ; write 8 elements starting at location 0 to file 1 of PLC 4
MAP DATA 2 7 6 3      ; write 3 elements starting at location 6 to file 7 of PLC 2
```

Half Duplex Mode

```
MAP DATA 4 1 0 8      ; write 8 elements starting at location 0 to file 1 of PLC 4 Station 4
MAP DATA 2 7 6 3      ; write 3 elements starting at location 6 to file 7 of PLC 2 Station 2
MAP DATA 5:2 7 6 3    ; write 3 elements starting at location 6 to file 7 of PLC 2 Station 5
```

SAMPLE TSP DOWNLOAD FILE

Assume that there are 2 Allen-Bradley PLC data maps defined. The first data map, MAP1, is used to communicate directly with 3 PLC slaves. The addesses of these PLCs will be 1, 2 and 3 for simplicity. The second data map, MAP2, is used to communicate with 2 additional PLCs that are connected to a 1770-KF2 module. The addresses of these PLCs will be 4 and 5 and the address of the 1770-KF2 modbule will be 10. Both data maps use the half duplex protocol. Although all 5 PLCs could have been defined in a single data map, they are setup in 2 data maps for illustrative purposes.

Now assume that task 1 will be used as the A-B PLC task and the file LINK1.RTU will be used to get a continuous update from all 5 PLCs. Polling can automatically begin whenever task 1 is started by including the line READ LINK1 in the START1.RTU file. Following is an example of what this LINK1.RTU file would look like:

```
; Allen-Bradley PLC Constant Poll File

map sel map1   ; sele first data map
map scan 1 0 0 16     ; scan 16 elements starting at 0 from file 0 of PLC 1
map scan 2 0 0 16     ; scan 16 elements starting at 0 from file 0 of PLC 2
map scan 2 1 0 16     ; scan 16 elements starting at 0 from file 1 of PLC 2
map scan 2 1 15 16    ; scan 16 elements starting at 15 from file 1 of PLC 2
map scan 3 7 0 8      ; scan 8 elements starting at 0 from file 7 of PLC 3
map scan 3 8 8 8      ; scan 8 elements starting at 8 from file 8 of PLC 3

map sel map2   ; sele second data map
map scan 10:4 0 0 16  ; scan 16 elements starting at 0 from file 10 of PLC 4
map scan 10:4 1 16 8  ; scan 8 elements starting at 16 from file 10 of PLC 4
map scan 10:5 7 0 8   ; scan 8 elements starting at 0 from file 10 of PLC 5
map scan 10:5 8 8 8   ; scan 8 elements starting at 8 from file 10 of PLC 5

if @ack(0) <> 0       ; if last scan command was unsuccessful  return   ; quit processing
file and constant poll
endif

map data 10:5 9 24 8  ; send 8 elements starting at 24 to file 10 of PLC 5

force 1 read link1    ; queue up msg to continue constant poll
```

Notice that the MAP SCAN and MAP DATA commands for MAP2 use the STN:DST notation where STN represents the 1770-KF2 address and DST represents the PLC address. Whenever data is requested from PLC 4 or PLC 5 it is read from the PLC's compatibility file. The compatibility file of each PLC is file number 10 which matches the address of the 1770-KF2 module. Similarly, whenever data is sent to PLC 4 or PLC 5 it is written to the same compatibility file.

To better understand how to exchange data between a PLC using a compatibility file and a TEST SCADA system, let's take a closer look at the two MAP SCAN commands shown above for PLC 5. These commands are:

```
map scan 10:5 7 0 8     ; scan 8 elements starting at 0 from file 10 of PLC 5
map scan 10:5 8 8 8     ; scan 8 elements starting at 8 from file 10 of PLC 5
```

Assume that file 7 is defined as an **Integer** file and file 8 is defined as a **Floating Point** file within PLC 5. Also assume that the data contained in files 7 and 8 is to be mapped with TSP channels as shown below:

	PLC 5	TSP Channel
File 7	Element 0	V1
	Element 1	V2
	Element 2	V3
	Element 3	V4
	Element 4	V5
	Element 5	V6
	Element 6	V7
	Element 7	V8
File 8	Element 0	A1
	Element 1	A2
	Element 2	A3
	Element 3	A4
	Element 4	A5
	Element 5	A6
	Element 6	A7
	Element 7	A8

Now assume that compatibility file 10 is defined as an **Integer** file within PLC 5 and the PLC is programmed to continuously copy the first 8 elements of files 7 and 8 into this file. The mapped data contained in file 10 might then look like this:

	PLC 5	TSP Channel
File 10	Element 0	V1
	Element 1	V2
	Element 2	V3
	Element 3	V4
	Element 4	V5
	Element 5	V6
	Element 6	V7
	Element 7	V8
	Element 8	upper 2 bytes of A1
	Element 9	lower 2 bytes of A1
	Element 10	upper 2 bytes of A2
	Element 11	lower 2 bytes of A2
	Element 12	upper 2 bytes of A3
	Element 13	lower 2 bytes of A3
	Element 14	upper 2 bytes of A4
	Element 15	lower 2 bytes of A4
	Element 16	upper 2 bytes of A5
	Element 17	lower 2 bytes of A5
	Element 18	upper 2 bytes of A6
	Element 19	lower 2 bytes of A6
	Element 20	upper 2 bytes of A7
	Element 21	lower 2 bytes of A7
	Element 22	upper 2 bytes of A8
	Element 23	lower 2 bytes of A8

File 7 and file 10 are both **Integer** files and each element takes up 2 bytes in either file. File 8 is a **Floating Point** file which requires 4 bytes per element. Therefore, each element in file 8 takes up the equivalent of 2 elements in file 10. In order to properly read all data mapped with TSP channels V1-V8 and A1-A8 from the compatibility file using the two MAP SCAN commands shown above, the TEST system data map for PLC ID 5 must be defined as shown below for files 7 and 8:

	PLC 5	TSP Channel
File 7	Element 0	V1
	Element 1	V2
	Element 2	V3
	Element 3	V4
	Element 4	V5
	Element 5	V6
	Element 6	V7
	Element 7	V8
File 8	Element 8	A1
	Element 9	A2
	Element 10	A3
	Element 11	A4
	Element 12	A5
	Element 13	A6
	Element 14	A7
	Element 15	A8

Again, as with PLC 5, define file 7 as an **Integer** file and file 8 as a **Floating Point** file in the TEST system. Notice that the elements in file 8 of the TEST system start at position 8 rather than 0. The reason for this should be clear after examining the two MAP SCAN commands in more detail. First of all, let's look at the command

```
map scan 10:5 7 0 8      ; scan 8 elements starting at 0 from file 10 of PLC 5
```

When the TEST SCADA system processes this command it first looks at the specified number of elements and the type of data contained in the specified file. Together, this information will determine how many bytes the TEST system will attempt to read from the PLC. In this command, the specified file is 7 which is defined as an **Integer** file. Since there are 2 bytes per element and the number of elements specified is 8, the TEST system will send out a command requesting 16 bytes from the PLC. The data will be read from the compatibility file, file 10, starting with element 0 as specified in the command. The returned data will be written to file 7 in the TEST data map, again starting with element 0 as specified in the command.

Now let's look at the second MAP SCAN command

```
map scan 10:5 8 8 8     ; scan 8 elements starting at 8 from file 10 of PLC 5
```

The specified file in this command is file 8 which is defined as a **Floating Point** file. Since there are 4 bytes per element and the command requests 8 elements from the PLC, the TEST system will send out a command requesting 32 bytes from the PLC. Again, the data will be read from compatibility file 10. However, instead of starting with element 0, data will be read starting with element 8 as specified in the command. The returned 32 bytes will be written to file 8 in the TEST data map starting with position 8.

TOTAL ENGINEERING SERVICES TEAM, INC.TEST INC.

TEST INC.

OFFICE	MAIN NUMBER	FAX NUMBER
New Orleans, LA	(504) 371-3000	(504) 371-3001
Lafayette, LA	(318) 269-0911	(318) 269-0910
Houston, TX	(713) 467-3113	(713) 467-8113
Ventura, CA	(805) 658-0403	(809) 658-9975
Anchorage, AK	(907) 276-5660	(907) 276-5361
Singapore	65-533-4108	65-534-2403
Bahrain	973-690-575	973-697-010

File: R1250.WP AMZ Printed April 3, 1996

Figure B.9
Interconnect Cable - 1784-CP10
PLC-5/11, -5/20, -5/30, -5/40, -5/40L, -5/60, -5/60L, -5/80 Processor to Terminal (using serial port)

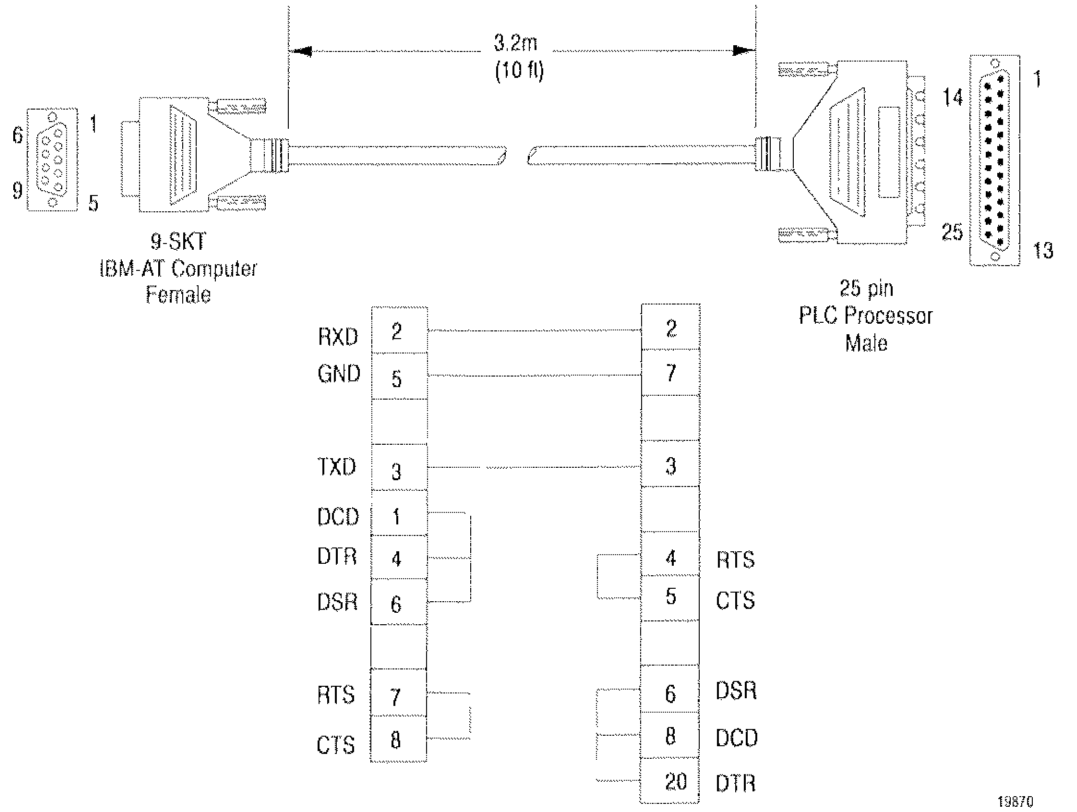
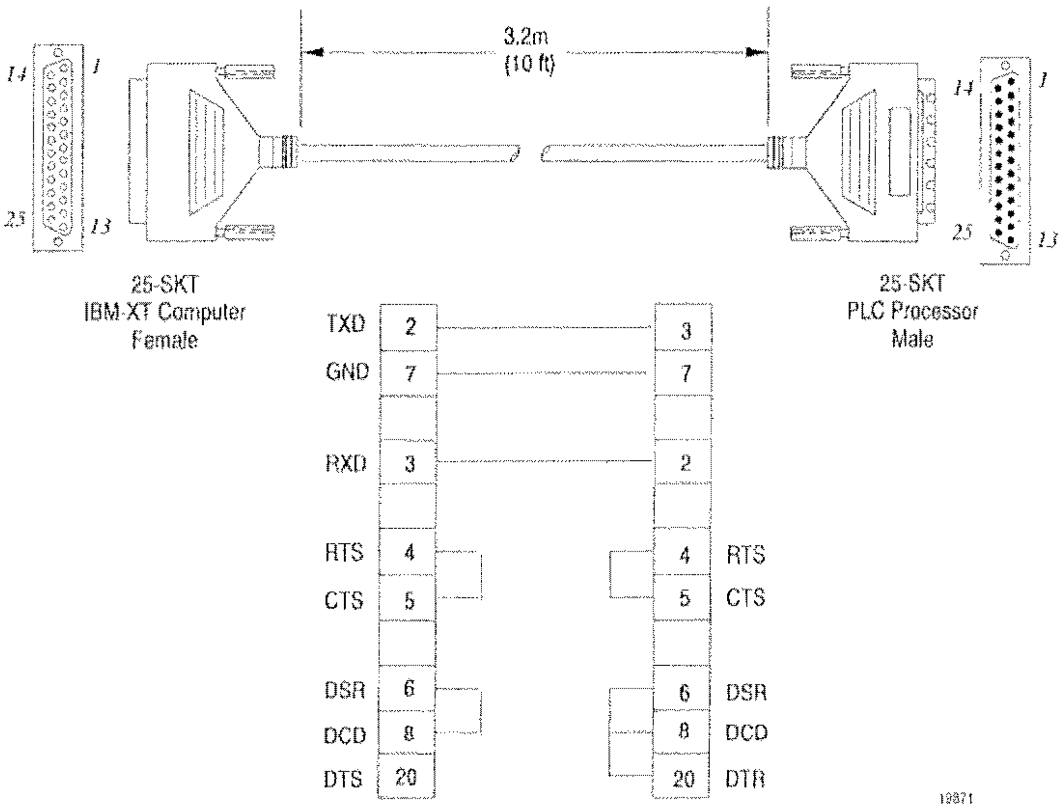


Figure B.10
Interconnect Cable - 1784-CP11
PLC-5/11, -5/20, -5/30, -5/40, -5/40L, -5/60, -5/60L, -5/80 Processor to Terminal (using a serial port)



If you are connecting a 1770-KF2 module to an RS-232-C compatible device (e.g., modem or computer), then you must mount the module within 50 cable feet of that device. For such applications, the module's GND must be connected to the GND of the modem or computer. This type of connection does not provide electrical isolation between the module and the connected device.

If a connection is made between the 1770-KF2 and an RS-422-A compatible device, you can mount the device and the module up to 4000 cable feet apart.

Direct Connection to a Computer

To connect the module directly to a computer, you can construct your own cable according to the wiring diagram in figure 3.6a. This cable plugs into the COMPUTER ASYNCHRONOUS connector on the module and the RS-232-C or RS-422-A compatible connector on the computer (figure 3.6a). Connect the cable shield at one end only. Be sure that the cable length does not exceed the RS-232-C limit of 50 feet or the RS-422-A limit of 4000 feet.

This type of connection includes the DTR signal to allow each end to detect the loss of the other end's ability to communicate. If your computer does not provide the DTR signal, jump pins 6 and 8 at the 1770-KF2 module to pin 20.

Figure 3.6a
Connection to a Computer

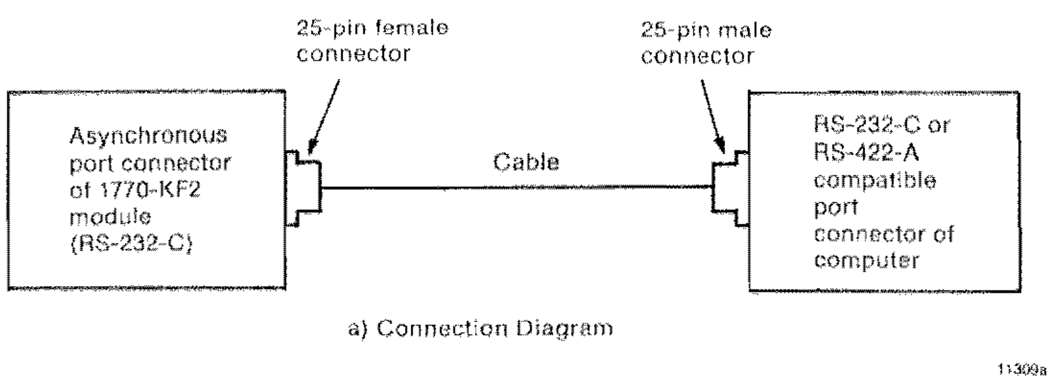


Figure 3.6b
Wiring Diagram for RS-232-C

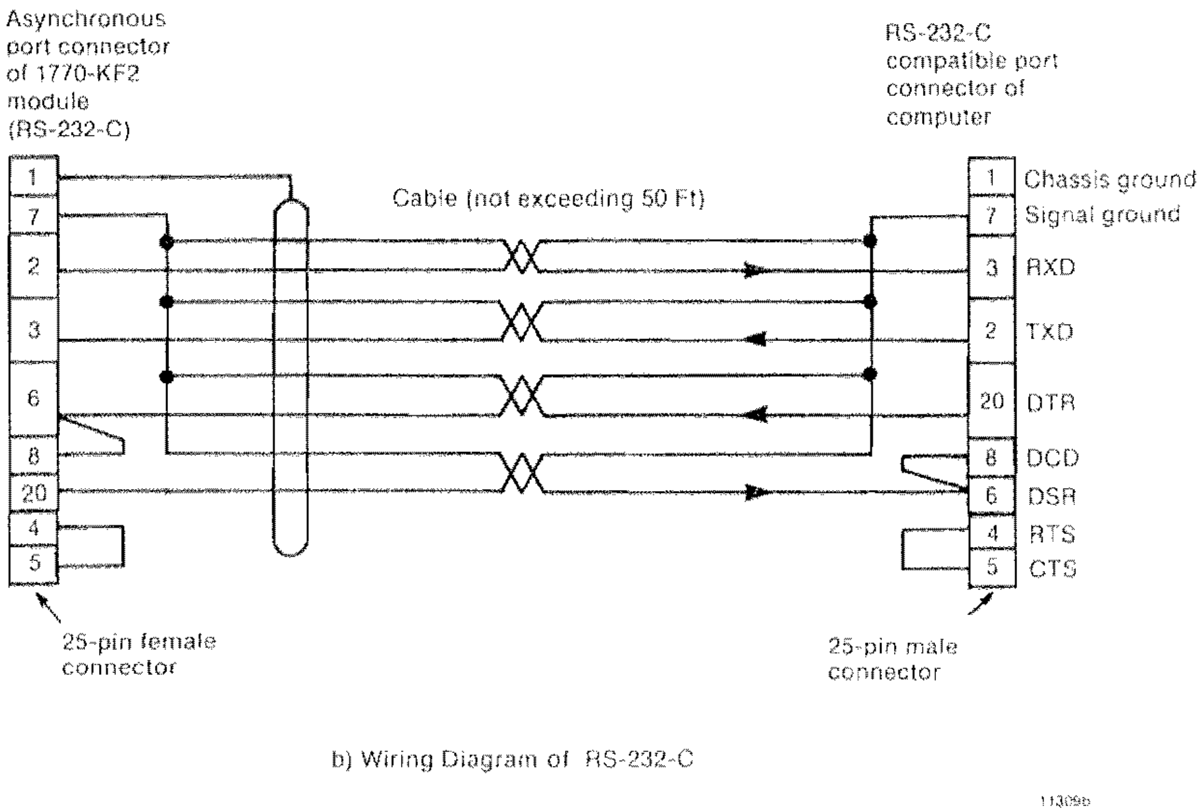
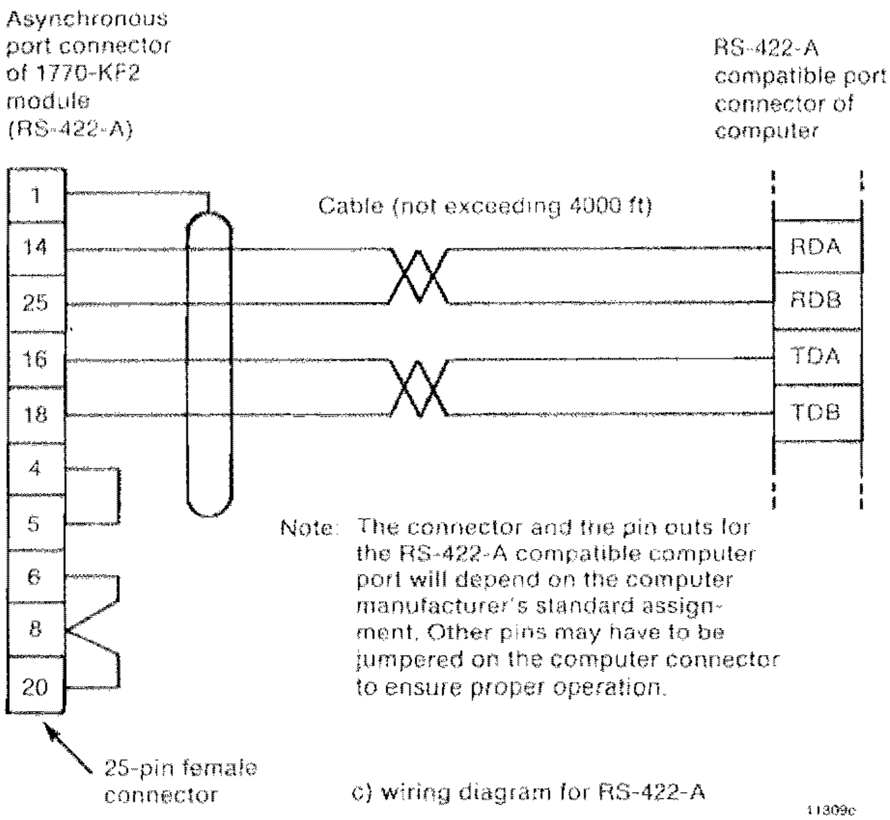


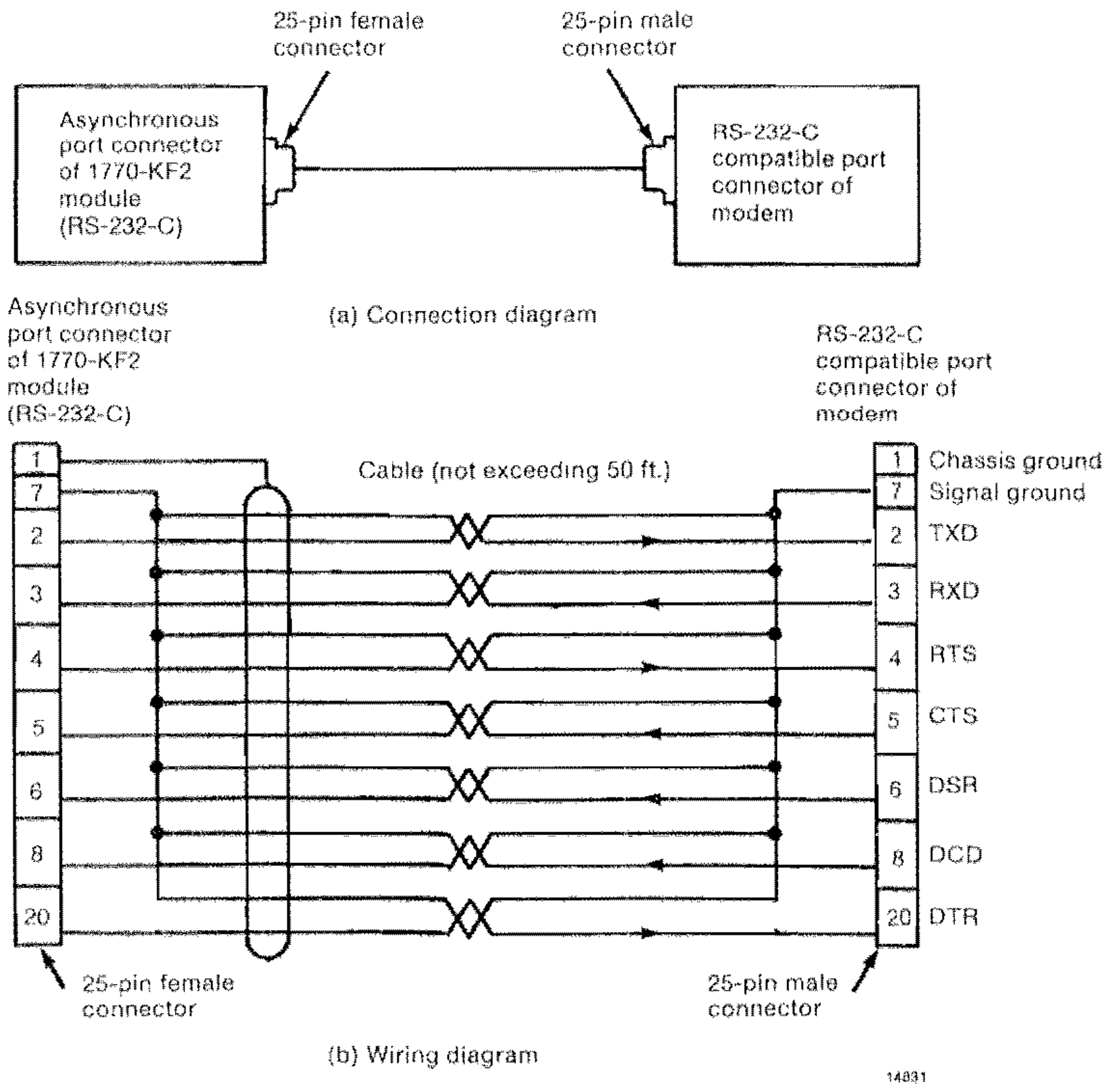
Figure 3.6c
Wiring Diagram for RS-422-A



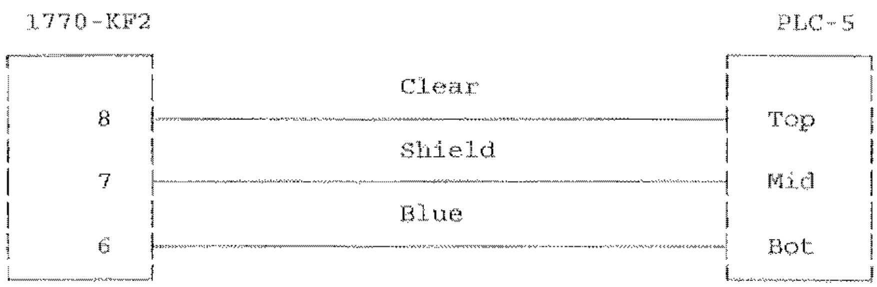
Connection to a Modem

To connect the module to a modem, you can construct your own cable according to the wiring diagram in figure 3.7. This cable plugs into the COMPUTER ASYNCHRONOUS connector on the module and the RS-232-C compatible connector on the modem (figure 3.7). Connect the cable shield at one end only. Be sure that the cable length does not exceed the RS-232-C limit of 50 feet.

Figure 3.7
Connection to a Modem



BLUE HOSE - 1770-KF2 to PLC-5

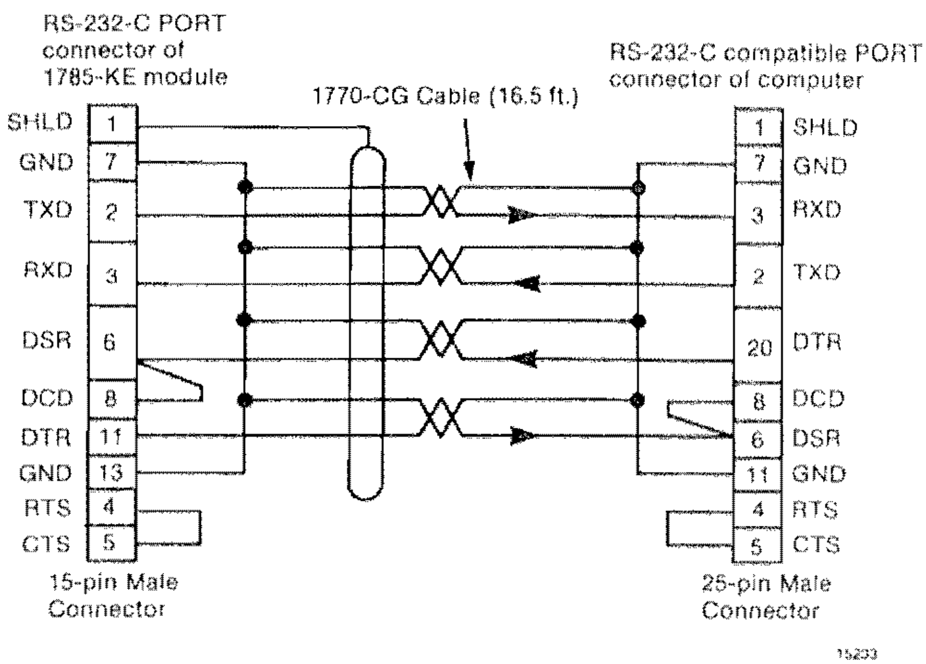


Direct Connection to a Computer

To connect the module directly to a computer, you can use a data terminal interface cable (cat. no. 1770-CG). This cable plugs into the RS-232-C PORT connector on the module and the RS-232-C compatible connector on the computer. Connect the cable shield at one end only.

The 1770-CG cable is 16.5 feet long. If you need a longer cable or a male/female adapter cable, you can construct your own according to the wiring diagram in Figure 4.2. Make sure that the cable length does not exceed 50 feet.

Figure 4.2
Wiring Diagram – RS-232-C PORT Connector to Computer



This type of connection includes the DTR signal to allow each end to detect the loss of the other end's ability to communicate. If your computer does not provide the DTR signal, jumper pins 6 and 8 at the module to pin 11.

Connection to Another Communication Module

You can connect the 1785-KE to another Data Highway interface module with a longline cable. This cable can be up to 7,000 feet long. However, remember that the cable length can limit the communication rate (refer to the section earlier in this chapter entitled **Electrical Characteristics of the RS-232-C Port**).

For information on how to construct a longline cable for connection to a:

- 1771-KG module, refer to figure 4.3
- 1773-KA or 1775-KA module, refer to figure 4.4

To construct the cable, use a male connector at each end. Use Belden 8723 or equivalent cable (available from Allen-Bradley under cat. no. 1778-CR). Connect the cable shield at one end only.

Figure 4.3
Connection to a 1771-KG Module

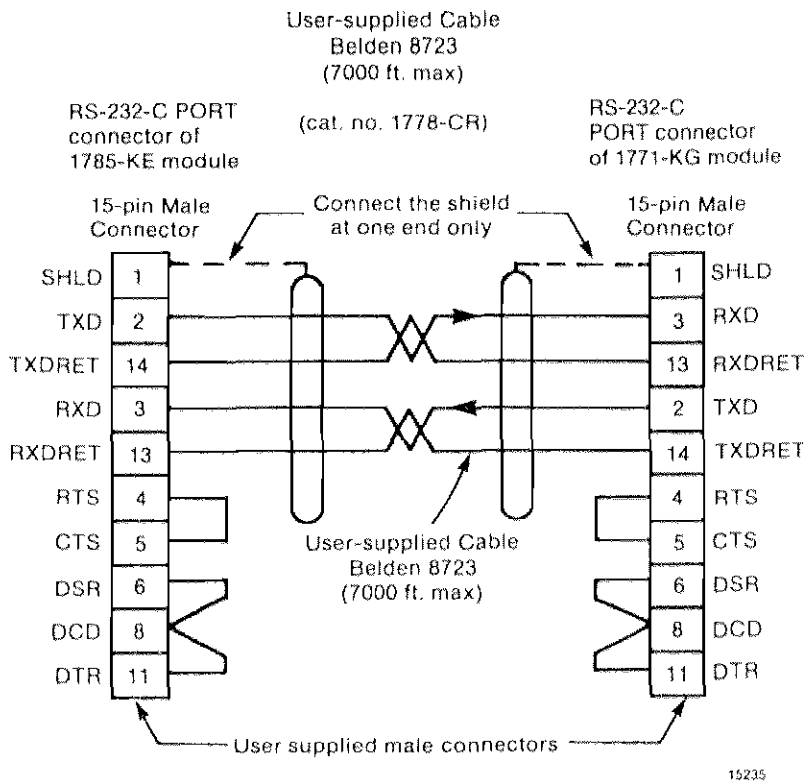
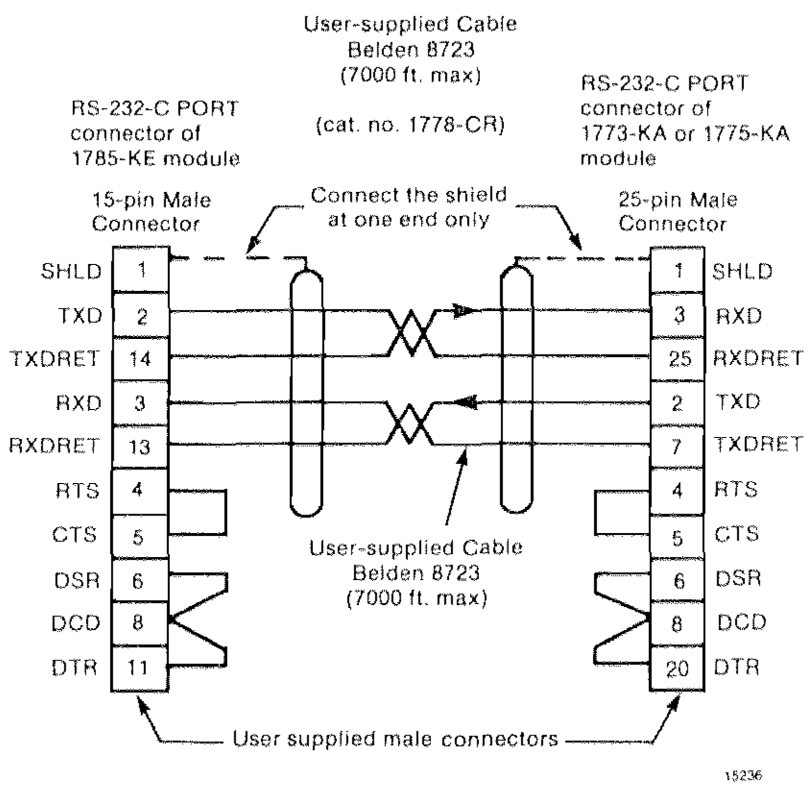


Figure 4.4
Connection to a 1773-KA or 1775-KA Module



Connection to a Modem

To connect the module to a modem, you can use the modem interface cable (cat. no. 1770-CP). This cable plugs into the RS-232-C PORT connector on the module and the RS-232-C compatible connector on the modem. Connect the cable shield at one end only.

The 1770-CP cable is 16.5 feet long. If you need a longer cable or a male/female adapter cable, you can construct your own according to the wiring diagram in Figure 4.5. Be sure that the cable length does not exceed the RS-232-C limit of 50 feet.

Figure 4.5
Wiring Diagram — RS-232-C PORT Connector to a Modem