Created:June 12, 1997
Modified:July 1, 1997

Gemini Controls Group Report

ICD 19: Allen-Bradley PLC Communications

Bret Goodrich and Stan Karewicz

ICD19/03

This Interface Control Document describes the EPICS communication to the Allen-Bradley PLC via serial and Remote I/O interfaces.

1.0 Introduction

1.1 Purpose

This document describes the interfaces between the EPICS crate and the Allen-Bradley PLC rack using either serial communications through a 1785-KE module or Remote I/O through the 1771-DCM and 6008-SV modules.

1.2 Scope

The audience for this document is EPICS programmers who need to communicate to an Allen-Bradley PLC program. This document describes the interface hardware, the communications protocol, and the EPICS programming. It does not describe Allen-Bradley ladder logic programming or Allen-Bradley PLC hardware not used for the purpose of communication with the EPICS crate.

1.3 Documents

The following documents are referenced in the text. They should be read when further information is required on a topic.

[1] *1785-KE Data Highway Plus Communications Interface Module*, Allen-Bradley, Publication 1785-6.5.2, May 1989.

[2] *Direct Communications Module User's Manual*, Allen-Bradley, Publication 1771-6.5.27, October 1987.

[3] *VMEbus Remote I/O Scanner User Manual*, Allen-Bradley, Publication 6008-6.5.11, September 1995.

[4] *EPICS Input Output Controller (IOC) Record Reference Manual*, Janet B. Anderson and Martin R. Kraimer, Argonne National Laboratory, December 1, 1994.

[5] *ICD 13-Standard Controller*, Bret Goodrich and Andrew Johnson, ICD13/03, March 12, 1997.

[6] *The Open Book: A Practical Perspective on Open Systems Interconnections*, Marshall Rose, Prentice-Hall, 1989.

[7] *PLC-5 Programming Software (Release 5.0)*, Allen-Bradley, Publication 6200-6.4.7, December 1994.

1.4 Abbreviations and Acronyms

A-B	Allen-Bradley
ai	analog input record
ao	analog output record
bi	binary input record
bo	binary output record
CRC	Cyclic Redundancy Check
DCM	Direct Communications Module
DH+	Data Highway Plus
EPICS	Experimental Physics and Industrial Control System
IOC	Input Output Controller
ISO	International Organization for Standardization
LANL	Los Alamos National Laboratory
mbbi	multi-bit binary input record
mbbo	multi-bit binary output record
OSI	Open Systems Interconnection
PLC	Programmable Logic Controller
RIO	Remote I/O
UAE	Universal Application Environment

1.5 Stylistic Conventions

All software commands typed by the use are printed in a **`Courier-Bold`** font

All software responses returned from a computer are printed in a `Courier` font.

All references to the documents are by number, i.e., [1].

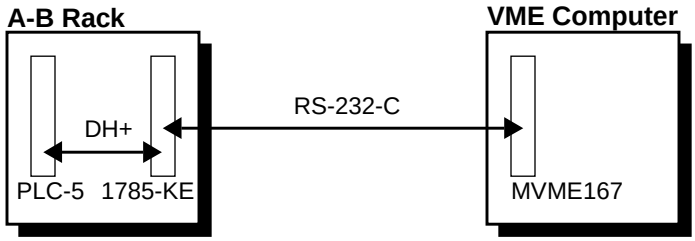
1.6 Revision History

- June 12, 1997, Creation.
- June 16, 1997, Added ISO-OSI model.
- July 1, 1997, Added Ladder Logic example.

Section I Serial Communications

The VME computer can communicate to the Allen-Bradley PLC through an A-B serial communications module located in an Allen-Bradley rack. This module, the 1785-KE, allows the VME computer to act as a node on the Data Highway Plus. Figure 1 shows the physical connections.

FIGURE 1. Serial Communications between the VME computer and the A-B PLC



The following discussions follow the ISO-OSI layering model for communications systems [6]. In this model, the layering is conceptual only, and a protocol can be implemented within a layer or across layers. The seven basic OSI layers are physical, data link, network, transport, session, presentation, and application.

2.0 Layers in the Serial Communications

The A-B serial communications software uses a collapsed version of the ISO-OSI model, since the DF1 driver software implements the network, transport, and session layers. Discussing EPICS device and record support within this model is somewhat problematical, since EPICS does not follow the OSI model.

2.1 Physical Layer

The three essential hardware elements needed for the PLC communications are the A-B 1785-KE module, the serial cable, and a serial port on a VME crate running EPICS.

2.1.1 1785-KE Module

The 1785-KE module is a communications interface that links an intelligent RS-232-C asynchronous device to an Allen-Bradley Data Highway Plus network. The module has an external connector for the Data Highway Plus, the RS-232, and a T50 Industrial Terminal. Various modes of operation can be set by adjusting the four switch assemblies on the 1785-KE module. Further description on these modes of operation, along with examples of applications can be found in the 1785-KE manual [1].

For communication with an EPICS crate, the configuration of the 1785-KE module must be set to the following:

- SW1
  - full duplex
  - CRC error checking
  - no parity
  - embedded responses
- SW2
  - station number should be uniquely selected
- SW3
  - DH+ 57.6 KBaud
  - RS-232 9600 Baud
  - local address
- SW4
  - reserved

2.1.2 VME Serial Port

Any serial port on a VME computer may be used to communicate with the A-B 1785-KE module, as long as this port has implemented standard VxWorks device support. Four serial ports are available on the Motorola MVME712-M transition module. These ports are female DB-25 connectors with hardware selectable DTE/DCE communications.

2.1.3 Serial Cable

The RS-232-C serial cable connects the 1785-KE serial port with the EPICS crate. The serial port on the 1785-KE is a female DB-15 connector, requiring a modified RS-232-C cable. Either an Allen-Bradley cable (1770-CG) can be purchased, or one made according to the specifications of the 1785-KE manual [1] on page 4-5, “Direct Connection to a Computer”. The cable should probably not exceed 50 feet, although this distance varies according to application and baud rate.

For applications which require a modem, either the Allen-Bradley cable (1770-CP) can be purchased, or one made according to the 1785-KE manual [1] on page 4-8, “Connection to a Modem”. Read the manual information about conforming modems before selecting equipment.

2.2 Data Link Layer

Communications between the 1785-KE module and the VME serial port are through the RS-232-C protocol, including full duplex and hardware handshaking. The implementation of the data link layer in the 1785-KE is beyond the scope of this document.

The implementation of the RS-232-C communications in the VME crate is provided by two software drivers: the specific serial port board driver and the EPICS serial driver **drvSerial**.

2.2.1 Serial Port Driver

VxWorks Serial port board drivers are be provided by board vendors or may need to be written by the application developer. Upon CPU initialization the driver installs in the

VxWorks operating system and will either create device names for its ports or provide a mechanism for the application developer to do so. The VxWorks command `iosDevShow` will print a list of available devices; some of these may be serial ports.

VxWorks provides serial port drivers for the Motorola MVME167 as part of the board support package. The drivers are implemented on the CPU's four serial ports as ports **/tyCo/0** through **/tyCo/3**. The console port uses **/tyCo/0**. All examples in this document refer to the 1785-KE communications port as **/ty/Co/1**

2.2.2 EPICS drvSerial

The EPICS serial port driver was implemented by Jeff Hill (LANL) for the Keck Observatory. This driver interfaces a specified serial port to the EPICS control environment. Because EPICS tasks are not allowed to block, this driver hides the complexities of non-blocking I/O. Input data is framed into individual messages, and the driver can determine if the serial link is down to retransmit data at a later time.

The EPICS serial driver does not have a convenient or simple interface for user applications. However for A-B communications, this complexity is hidden by another driver layer, **drvAbDf1**, described in the following section. To use the driver for A-B communications, it must be added to the local EPICS configuration, compiled in an EPICS environment such as UAE, then loaded onto the IOC at boot time.

2.3 Network Layer

The DF1 protocol provides the network, transport, and session layers of the OSI layers. The protocol routes data between nodes (network), reliably transmits messages (transport), and establishes and maintains connections between the nodes (session).

2.3.1 EPICS drvAbDf1

The EPICS A-B DF1 driver was implemented by Jeff Hill (LANL) for the Keck Observatory. This driver transmits and receives ANSI F1 full duplex messages between the 1785-KE and VME computer. The messages are formatted to be retransmitted by the 1785-KE module onto the Data Highway Plus, and received by the requested PLC module.

The driver operates by reading a block of data from the PLC at a fixed rate of 8 Hz. Data being read from the PLC to the EPICS database is then transferred to the appropriate record and **I/O INTR** scanning is triggered if needed. Data to be written to the PLC is then transmitted to the 1785-KE for retransmission.

The driver initialization must have the source node address of the 1785-KE, and the size of the read and write memory buffers. The source node address is used in all data messages to identify the DH+ initiator. The read and write buffer sizes are used to perform block transfers of the PLC memory for performance reasons. The sizes must be less than or equal to the memory size specified in the PLC module.

The command format used for the data transmission across the serial line uses the PLC-2 type commands. These type of commands place some constraint upon the presentation of the PLC data to the VME computer. Most importantly, the PLC data to be transferred must be in one PLC file with the same file number as the 1785-KE

module node address. Thus, if the 1785-KE node address is 0x20, the PLC program must have a compatibility file number 0x20 with the required data. The use of PLC-3 type commands, which the PLC-5 module supports, would remove this restriction, but has not been implemented in the EPICS A-B DF1 driver.

The following examples of DF1 transactions have been lifted from the 1785-KE module manual [1]. In these examples, the computer (or 1785-KE) node address is 0x29 and the PLC-5 node address is node address 0x20. Note that these examples use BCC error checking, while the EPICS driver uses CRC error checking.

2.3.2 Writing to the PLC

This example writes four words, or eight bytes, of data from the computer to the PLC-5 using the PLC-2 type command Unprotected Write:

FIGURE 2. PLC-2 Unprotected Write

1. The computer sends the command to the 1785-KE:

```
DLE STX DST SRC CMD STS TNS TNS ADR ADR ----------- DATA--------------- DLE ETX BCC
10  02  29  20  08  00  44  01  28  00  22  11  44  33  66  55  88  77  10  03  DE
```

2. The 1785-KE responds to the computer

```
DLE ACK
10  06
```

3. The 1785-KE sends the command to the PLC-5:

```
The 1785-KE sends the command onto the DH+ and the PLC-5 node at 0x29.
```

4. The PLC-5 sends the reply to the 1785-KE:

```
The PLC-5 executes the command, formats a reply, and sends it back to the 1785-KE.
```

5. The 1785-KE sends the PLC-5 reply back to the computer:

```
DLE STX DST SRC CMD STS TNS TNS DLE ETX BCC
10  02  20  29  48  00  44  01  10  03  2A
```

6. The computer responds to the 1785-KE:

```
DLE ACK
10  06
```

2.3.3 Reading from the PLC

This example reads the four words written above back from the PLC to the computer using the PLC-2 type command Unprotected Read:

FIGURE 3. PLC-2 Unprotected Read

1. The computer sends the command to the 1785-KE:

```
DLE STX DST SRC CMD STS TNS TNS ADR ADR SIZ DLE ETX BCC
10  02  29  20  01  00  45  01  28  00  08  10  03  40
```

2. The 1785-KE responds to the computer:

```
DLE ACK
10  06
```

3. The 1785-KE sends the command to the PLC-5:

```
The 1785-KE sends the command onto the DH+ and the PLC-5 node at 0x29.
```

4. The PLC-5 sends the reply to the 1785-KE:

```
The PLC-5 executes the command, formats a reply, and sends it back to the 1785-KE.
```

5. The 1785-KE sends the PLC-5 reply back to the computer:

```
DLE STX DST SRC CMD STS TNS TNS ------------DATA--------------- DLE ETX BCC
10  02  20  29  48  00  44  01  22  11  44  33  66  55  88  77  10  03  2A
```

6. The computer responds to the 1785-KE:

```
DLE ACK
10  06
```

2.3.4 Ladder Logic Example

The PLC program rungs (see [7]) in Figure 4 are not necessary to have communication working. They have only been used for display purposes on the PLC end. Communication will work with no PLC program as long as the PLC has a data file corresponding to the 1785-KE DH+ node address, and the size of this file matches or exceeds the size specified in the EPICS read and write buffers.

FIGURE 4. Ladder Logic Example

```
Thu Jun 19, 1997   Page 1
Program Listing Report              PLC-5/20    File KEDEMO             Rung 2:0


Rung 2:0
1785-KE DH+ address =13 octal = 11 decimal, this determines PLC file number
accessed through serial port/DH+ link (PLC2 message format).
In this case, VME system is writing into word N11:23.
This rung is for display purposes only. MOV instruction allows to observe bit
patterns on the display of the output module, located in Rack 0, slot 1.
|                                                          +MOV---------------+ |
+----------------------------------------------------------+MOVE              +-+
|                                                          |Source      N11:23| |
|                                                          |                 32| |
|                                                          |Destination  O:001| |
|                                                          |                 32| |
|                                                          +------------------+ |
N11:23
     -MOV-  2:0 2:1
O:001
     -MOV-  2:0
```

```
Rung 2:1
This rung can be used for verification of what is being written to PLC memory.
Value in word N11:23 is transferred into word N11:13, which can be, in turn,
read back by VME via 1785-KE.
And again, MOV to O:04 is used for display purposes.
|                                                          +MOV---------------+  |
+----------------------------------------------------------++MOVE              ++-+
|                                                          ||Source      N11:23|| |
|                                                          ||                 32|| |
|                                                          ||Destination N11:13|| |
|                                                          ||                 32|| |
|                                                          |+------------------+| |
|                                                          |+MOV---------------+| |
|                                                          ++MOVE              ++ |
|                                                           |Source      N11:13|  |
|                                                           |                 32|  |
|                                                           |Destination  O:004|  |
|                                                           |                 32|  |
|                                                           +-------------------+  |
N11:13
     -MOV-  2:1 2:1
N11:23
     -MOV-  2:0 2:1
O:004
     -MOV-  2:1

Rung 2:2
|                                                                                  |
+-------------------------------[END OF FILE]-----------------------------------+
|                                                                                  |
```

2.4 Presentation Layer

The presentation layer of the PLC communications interfaces the EPICS records with the network layer support provided by **drvAbDf1**. This layer is implement through EPICS device support for six standard records: analog input (*ai*), analog output (*ao*), binary input (*bi*), binary output (*bo*), multi-bit binary input (*mbbi*), and multi-bit binary output (*mbbo*).

2.4.1 Common EPICS Record Fields

The following fields are common to most or all the above records. How they are used are also similar.

DTYP. The Allen-Bradley serial communications device support (DTYP) is "**AB DF1 serial**".

SCAN. The scanning mechanism (SCAN) for *ai* device support can use **I/O Intr** along with **Passive**, **Event**, or any defined scan rate. When processed using I/O Intr the record will process on every change the drvAbDf1 driver detects in the value of the word or bit.

INP or OUT. The input (INP) or output (OUT) strings for the record identify the serial port, destination node number, and word number for the PLC word to be read or written. An example string for *ai*, *ao*, *mbbi*, or *mbbo* records would be:

```
@/tyCo/1 41 23
```

For *bi* or *bo* records the string also contains the bit number to read or write:

```
@/tyCo/1 41 23 5
```

Note the leading '@' character identifying this string as a link type of INST_IO. This string is fixed for each record instantiation.

2.4.2 Analog Input Record

The *ai* record returns a double-precision value to the EPICS database from the underlying device support routines. Because the A-B serial device support reads words from the hardware some conversion method must be used.

LINR. The linearization field (LINR) converts between the RVAL and VAL fields. Conversion types may be "**No Conversion"**, "**Linear"**, or a user-defined breakpoint table. A conversion method of "**No Conversion**" will simply typecast the RVAL field into the VAL field. A "**Linear**" conversion method requires values for ESLO and EGUL. See the EPICS Record Reference Manual [4] for more information on *ai* record conversions.

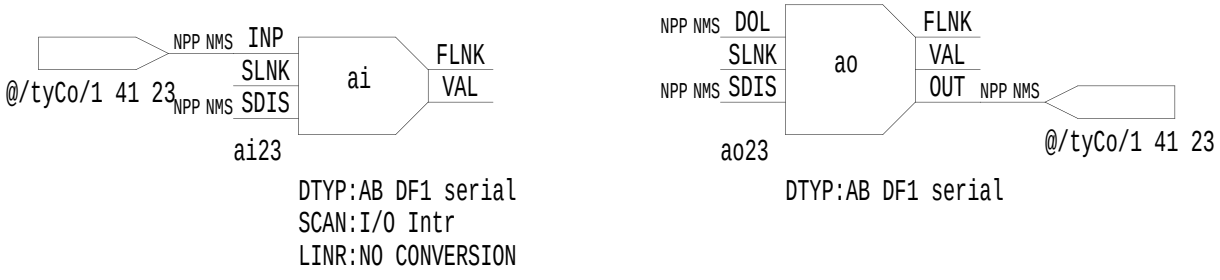
2.4.3 Analog Output Record

The *ao* record writes a double-precision value from the EPICS database to the underlying device support. Because the A-B serial device support writes words to the hardware some conversion method must be used.

LINR. Conversion of the *ao* record's VAL field to the output RVAL value is similar to the *ai* record conversion described in "Analog Input Record" on page 9.

Figure 5 shows an *ai* record reading word 23 from PLC node 41 on serial port /tyCo/1 and an *ao* record writing to the same word.

FIGURE 5. *ai* and *ao* record using A-B serial device support



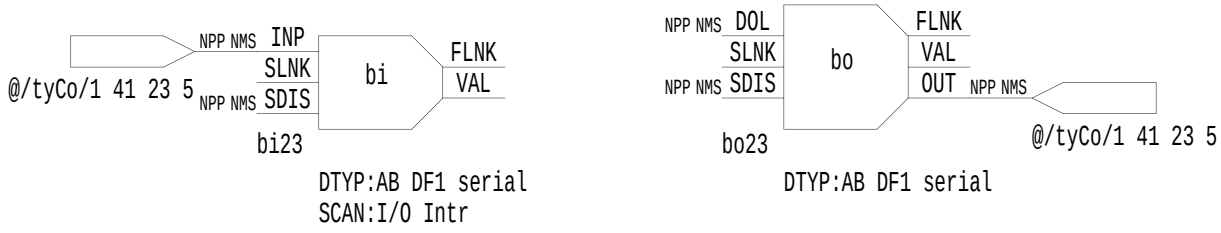
2.4.4 Binary Input Record

The *bi* record reads one bit into the EPICS database from the underlying device support routines. The value is enumerated into "**enabled**" or "**disabled**" states, and for A-B serial device support, "**enabled**" means the bit value is low.

2.4.5 Binary Output Record

The *bo* record writes one bit from the EPICS database into the underlying device support routines. The value is enumerated into "**enabled**" or "**disabled**" states, and for A-B serial device support, "**enabled**" means the bit value is low.

FIGURE 6. *bi* and *bo* record using A-B serial device support



2.4.6 Multi-bit Binary Input Record

The *mbbi* record reads a specified number of bits into the EPICS database from the underlying device support routines. The input bit pattern is converted into an enumerated type in the VAL field using a lookup table of bit pattern to index conversion.

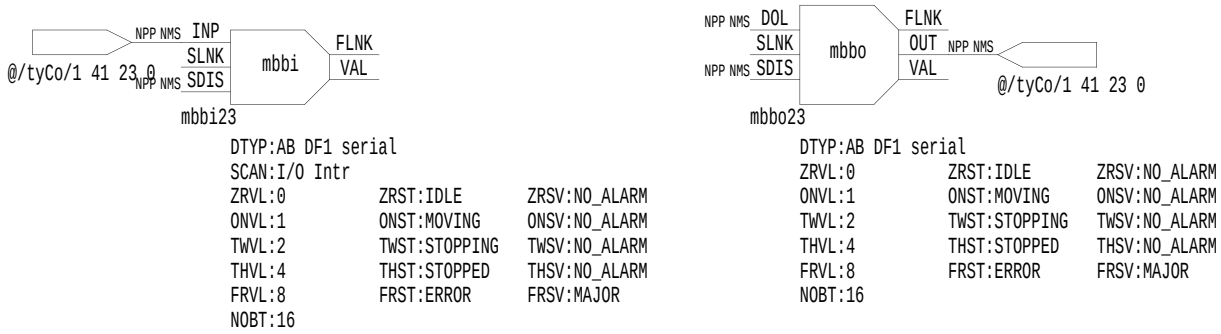
NOBT. The number of bits field (NOBT) must be set to the size of the data read. In the case of the A-B serial device support, this is a maximum of 16 bits.

2.4.7 Multi-bit Binary Output Record

The *mbbo* record writes a specified number of bits from the EPICS database into the underlying device support routines. The enumerated type VAL field is converted to an output bit pattern using a lookup table of index-to-bit pattern conversion.

NOBT. The number of bits field (NOBT) must be set to the size of the data read. In the case of the A-B serial device support, this is a maximum of 16 bits.

FIGURE 7. *mbbi* and *mbbo* record using A-B serial device support



2.5 Application Layer

An EPICS database provides the application layer of the OSI model. Such a database would be the Enclosure Control System communications with the A-B rack controlling the carousel motion.

3.0 Installation of Serial Communications Software

The A-B serial communications software is not part of the standard EPICS distribution, nor of the Gemini EPICS distribution. Applications which require this software should install it in their particular application environment.

The following installation instructions are for the EPICS UAE environment.

3.0.1 ascii

The UAE **ascii** directory contains the subdirectory **cat_ascii** to catenate new ASCII description files onto the base EPICS release. The A-B serial communications software contains no new records, one new device, and two new drivers.

- Change directory to **ascii/cat_ascii**.
- Edit or create the file **devSup.ascii** and add the following lines:

```
! Allen Bradley DF1
“ai”            INST_IO         “devAiAbDf1”        “AB DF1 serial”
“ao”            INST_IO         “devAoAbDf1”        “AB DF1 serial”
“bi”            INST_IO         “devBiAbDf1”        “AB DF1 serial”
“bo”            INST_IO         “devBoAbDf1”        “AB DF1 serial”
“mbbi”          INST_IO         “devMbbiAbDf1”      “AB DF1 serial”
“mbbo”          INST_IO         “devMbboAbDf1”      “AB DF1 serial”
```

- Edit or create the file **drvSup.ascii** and add the following lines:

```
“drvSerial”
“drvAbDf1”
```

3.0.2 src

There are three source code files and two include files to add to the src directory: **devAbDf1.c**, **drvAbDf1.c**, **drvAbDf1.h**, **drvSerial.c**, and **drvSerial.h**.

- Change directory to src.
- Add the files **devAbDf1.c**, **drvAbDf1.c**, **drvSerial.c**, **drvAbDf1.h**, and **drvSerial.h**.
- Modify **Makefile.Vx** and add the first three file names to the definition of SRCS.c.

3.0.3 startup

The object code file names for the above files must be added to the VxWorks startup script. The definitions of three VxWorks variables must be added following the object code file names.

- Edit **startup.vws**.
- Add the following lines in the record, device, driver, *etc.* section after the EPICS code has been loaded but before **iocInit** is executed.

```
ld < $(version)/bin/$(arch)/x.o
ld < $(version)/bin/$(arch)/drvSerial.o
ld < $(version)/bin/$(arch)/drvAbDf1.o
ld < $(version)/bin/$(arch)/devAbDf1.o
```

- Immediately following, add the definitions for the VxWorks variables for the 1785-KE node address, and the read and write block sizes. The first variable must be the same as the address set on SW2 on the 1785-KE module. Change the values in the example to your configuration.

```
drvAbDf1SrcStationNumber = 11;
drvAbDf1BlockReadDataSize = 244;
drvAbDf1BlockWriteDataSize = 244;
```

4.0 Problems with Serial Communications Software

The known problems with the A-B serial communications software are listed below. Some are for informational purposes only.

4.0.1 Reboot

If the software has successfully linked with the 1785-KE module and is then rebooted, it crashes after **iocInit** with an assertion failure. A second reboot will succeed. Then problem seems to be linked to the reception of a DF1 message before the software is ready to handle it.

4.0.2 PLC-2 type unprotected writes

The original source code has been modified by the Gemini Project to use unprotected writes rather than protected writes. The original source code communicated with 1771-KG and PLC-2 modules, rather than with the 1785-KE and PLC-5/20 modules of the Gemini Project. The PLC-5 does not accept protected write commands.

4.0.3 PLC-2 type addresses

The PLC-5 module accepts PLC-2 type addresses, as long as the processor has a compatibility file with the same node number as the 1785-KE. Only memory inside this

data is accessible using the PLC-2 type addresses. The software currently uses only the PLC-2 type.

4.0.4 Error messages

Error messages from the drivers have not been installed into the VxWorks error message symbol table. The serial driver drvSerial uses error code 1000 and the drvAbDf1 uses error code 1001, although either of these can be changed in the include files if a conflict occurs.

- **drvSerial** error messages are:

```
#define M_drvSerialLib (1000<<16)
#define S_drvSerial_OK 0
#define S_drvSerial_noEntry (M_drvSerialLib | 1)        /* no response on the queue */
#define S_drvSerial_badParam (M_drvSerialLib | 2)       /* unable to set option */
#define S_drvSerial_paramConflict (M_drvSerialLib | 3)  /* requested options
                                                         * conflict between
                                                         * applications */
#define S_drvSerial_noInit (M_drvSerialLib | 4)         /* init driver not called */
#define S_drvSerial_linkInUse (M_drvSerialLib | 5)      /* serial device is in use */
#define S_drvSerial_noParser (M_drvSerialLib | 6)       /* no parser */
#define S_drvSerial_EOF (M_drvSerialLib | 7)            /* link down */
#define S_drvSerial_OVF (M_drvSerialLib | 8)            /* no msg term before buf ovf */
#define S_drvSerial_queueFull (M_drvSerialLib | 9)      /* queue quota exceeded */
#define S_drvSerial_invalidArg (M_drvSerialLib | 10)    /* invalid argument */
#define S_drvSerial_noDevRead (M_drvSerialLib | 11)     /* open for read failed */
#define S_drvSerial_noDevWrite (M_drvSerialLib | 12)    /* open for write failed */
#define S_drvSerial_linkDown (M_drvSerialLib | 13)      /* serial link is down */
#define S_drvSerial_noMemory (M_drvSerialLib | 14)      /* out of dynamic memory */
#define S_drvSerial_noneAttached (M_drvSerialLib | 15)  /* no app is using the link */
#define S_drvSerial_internal (M_drvSerialLib | 16)      /* internal */
```

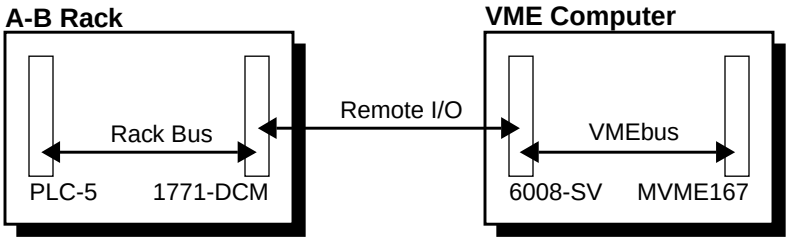
- **drvAbDf1** error messages are:

```
#define M_drvAbDf1Lib (1001<<16U)
#define S_drvAbDf1_OK 0
#define S_drvAbDf1_badParam (M_drvAbDf1Lib | 1)         /* bad parameter */
#define S_drvAbDf1_noInit (M_drvAbDf1Lib | 2)           /* init driver not called */
#define S_drvAbDf1_linkDown (M_drvAbDf1Lib | 3)         /* serial link is down */
#define S_drvAbDf1_noMemory (M_drvAbDf1Lib | 4)         /* out of dynamic memory*/
#define S_drvAbDf1_badFrame (M_drvAbDf1Lib | 5)         /* corrupt input frame */
#define S_drvAbDf1_abError (M_drvAbDf1Lib | 6)          /* Allen Bradley error */
```

Section II Remote I/O Communications

The VME computer can communicate to an A-B PLC through a Remote I/O (RIO) module located in an A-B rack. This module, the 1771-DCM, allows the VME computer to act as node on an RIO network, usually with the 1771-DCM module as the only other node. Figure 8 shows the physical layout of this communications system.

FIGURE 8. Remote I/O communications between the VME computer and the A-B PLC



5.0 Layers in the RIO Communications

[This section is TBD.]

6.0 Installation of the RIO Communications Software

[This section is TBD.]

7.0 Problems with the RIO Communications Software

[This section is TBD.]