Twinax OEM Interface Specification for Windows 95/NT

07/28/00

Owner: Paul R. Sappie
Host Integration Solutions Development Manager
Internet: sappie@us.ibm.com
Notes Mail: Paul Sappie/Raleigh/IBM@IBMUS

PComm Contact: Bill Gaither
DLC Development
Internet: gaither@us.ibm.com
Notes Mail: Bill Gaither/Raleigh/IBM@IBMUS

CA/400 Contact: Mark R. Vanderwiel
Client Access Communications Win32
Internet: vanderwl@us.ibm.com
Notes Mail: Mark Vanderwiel/Rochester/IBM@IBMUS

IBM Corporation
F78A/502
P.O. Box 12195
Research Triangle Park,
NC 27709 U.S.A.

Attention

This document is a draft version and is subject to change without any notification. The latest version of the document is available through the Personal Communications "Application programming interfaces" page at: http://www-4.ibm.com/software/network/pcomm/about/api.

Revision history is:

Change Date	Revision Level	Revision Code	Description
03/07/97	Draft	0.00	Original Text
05/30/97	Draft	1.00	Added Windows 95 interface
08/30/97	Draft	1.10	Added IBM compatible PnP adapter support
06/28/2000	Draft	1.11	Update title page.

Contents

Figures

1.0 Introduction

This document describes the Twinax OEM interface for IBM Personal Communications Version 4.1 for Windows NT, IBM eNetwork Personal Communications Version 4.2 for Windows 95 and Windows NT, IBM Communications Server for Windows NT Version 5.01, and IBM Client Access for Windows 95 and Windows NT (Version 3 Release 1 Mod 3) products. The Twinax OEM Interface package includes a C function prototype/header file and two platform specific run-time library files. The OEM device driver objects should be linked with the library for its platform. The library code provides a set of functions to set up the function table used to communicate with the IBM products, and to call the IBM products using the function table. This interface specification defines APPC (and SNA pass-through) and Console connection types. However, the console connection type is only supported by the IBM Personal Communications Version 4.2 for Windows 95 and Windows NT.

Note: This document follows Intel bit assignment philosophy (bit0 = LSP, bit7 = MSB).

2.0 Design Overview

2.1 Windows NT

The Twinax lowest level kernel-mode driver is divided into two parts (see Figure 1 on page 2). The upper layer driver is the Twinax Common Driver provided by IBM and the lower layer driver is the Twinax MAC driver provided by an OEM vendor. At first, the common driver sets the function table addresses in an IRP's input buffer and sends it to the OEM MAC driver by using the device name "TwinaxOEM". When the OEM MAC driver processes the IRP, the common driver's function addresses are saved for a direct call, and the MAC driver's function addresses are set in the IRP's output buffer in order for the common driver to get the function addresses for a direct call. After the function tables are filled with the actual function addresses, both the drivers are able to make a direct call.

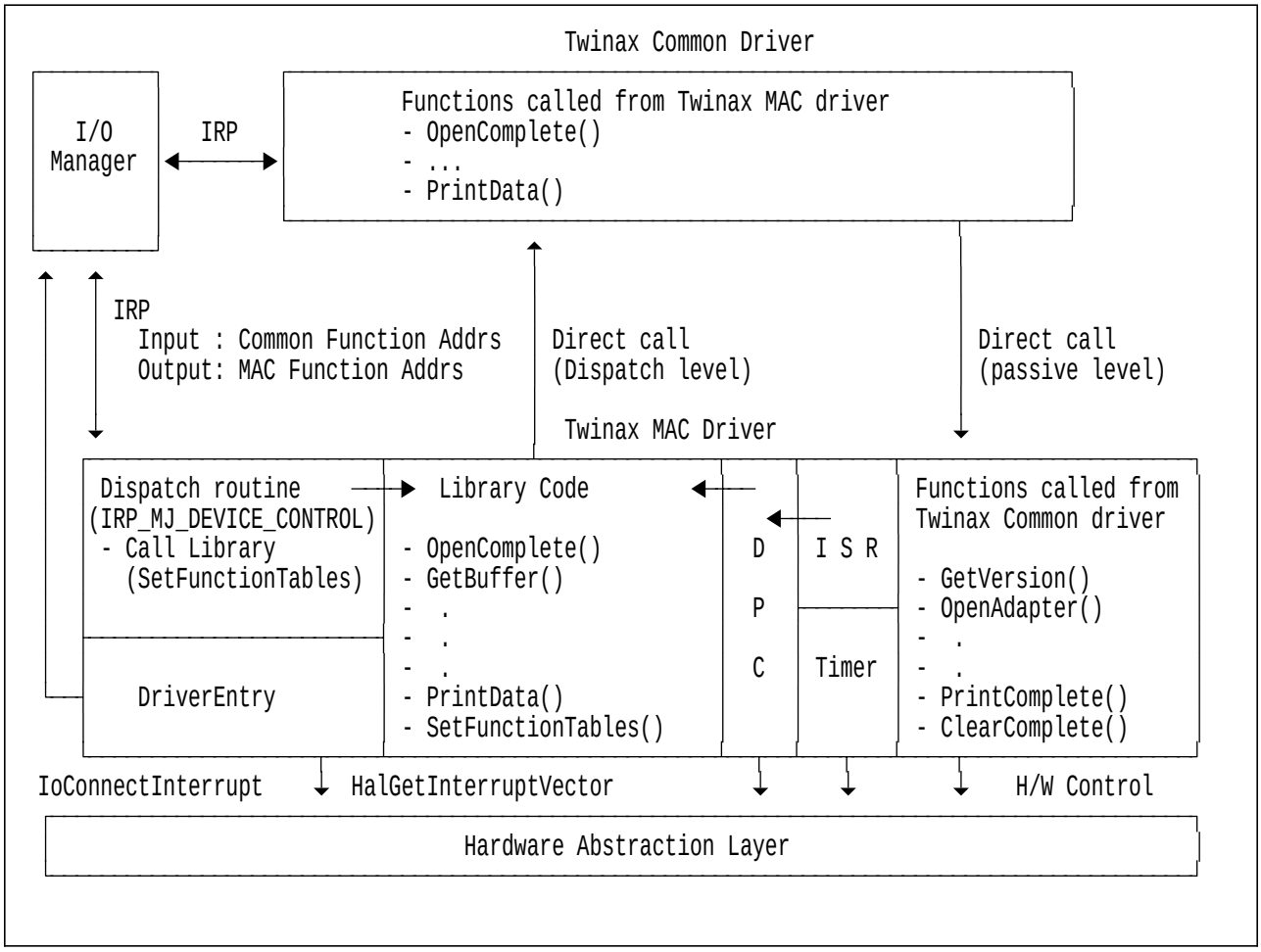

Figure 1. Twinax kernel-mode device drivers for Windows NT

2.2 Windows 95

The Twinax lowest level virtual device driver is divided into two parts (see Figure 2 on page 3). The upper layer driver is the Twinax Common Driver provided by IBM and the lower layer driver is the Twinax MAC driver provided by an OEM vendor. At first, the common driver loads the OEM driver by reading the string value of the HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Twinax\DeviceName registry key, then sets its function addresses in an IOCTL's input buffer and makes eSetFunctionTable (0x95) IOCTL call to the OEM MAC driver. When the OEM MAC driver processes the IOCTL, the common driver's function addresses are saved for a direct call, and the MAC driver's function addresses are set in the IOCTL's output buffer in order for the common driver to get the function addresses for a direct call. After the function tables are filled with the actual function addresses, both the drivers are able to make a direct call.

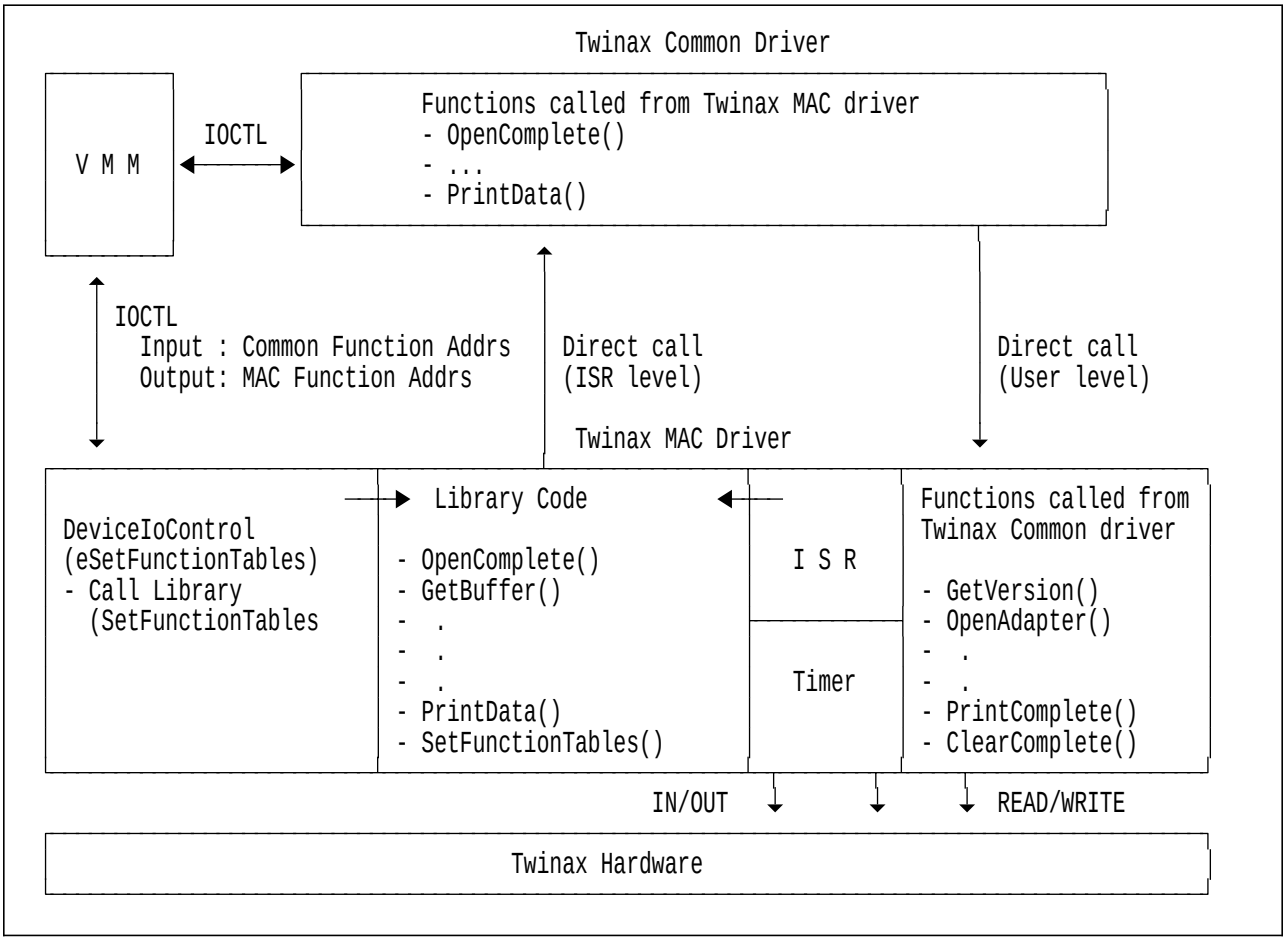

Figure 2. Twinax virtual device drivers for Windows 95

3.0 Interface Details

3.1 IRP Interface for Windows NT

The function tables used for a direct call between the Common Driver and the MAC Driver are set by the **SetFunctionTables** function of the MAC library. The MAC driver should call this function when an IRP for device I/O Control (IRP_MJ_DEVICE_CONTROL) is processed in the DriverEntry routine.

3.2 IOCTL Interface for Windows 95

The function tables used for a direct call between the Common Driver and the MAC Driver are set by the **SetFunctionTables** function of the MAC library. The MAC driver should call this function when an I/O control code for eSetFunctionTables (0x95) is processed in Win32DeviceIoControl routine.

3.3 Call Interface

3.3.1 MAC Driver functions

The common driver directly calls the MAC driver functions at the passive or user level by using the function table addresses.

Get Version

```
ULONG GetVersion (
                    ULONG *pVersion,
                    BYTE    *pLevel
                                    );
```

The **GetVersion** function is used to make sure that the MAC driver's version and support level are correct. The driver's version represents the version of the interface specification. When the interface is enhanced, the new interface version will be defined. The support level presents the connection type which the OEM MAC driver supports.

Parameters

- **pVersion**: Points to the storage in which to return MAC version

  `0x02 = Version of the MAC driver which follows the initial OEM interface`

- **pLevel** : points to the storage in which to return MAC support level

```
xx = The bits returned are defined as follows

     7  6  5  4  3  2  1  0
     -  -  -  -  -  -  -  -
     |  |  |  |  |  |  |  |
     |  |  |  |  |  |  |  APPC (5150-001)
     |  |  |  |  |  |  SBCS Display (3197-C1)
     |  |  |  |  |  SBCS Printer (5224-001, 5256-002, 5219-D01)
     |  |  |  |  DBCS Display (5555-C01)
     |  |  |  DBCS Printer (5553-B01)
     |  |  Reserved for future enhancement
     |  Reserved for future enhancement
     SNA Pass-through (5150-001)
```

Return Value

The **GetVersion** function returns the following values:

- 0x00 = Normal Return Code

Connection Type

The **GetVersion** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

Open Adapter

```
ULONG OpenAdapter (
                    ULONG adapterNumber,
                    ULONG adapterType,
                    ULONG irq,
                    ULONG portAddress,
                    ULONG memoryAddress
                                        );
```

The **OpenAdapter** function is used to open one adapter. All initialization needed for opening an adapter must be done by the MAC driver in this function. If an adapter cannot be initialized for some reason, the MAC driver should return an appropriate error code. The error code will be mapped to a product message and it will be logged/displayed.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

- **adapterType**

```
0x00        : OEM Adapter
0x01 - 0xFF : Reserved
```

- **irq** (Not used for OEM adapters)
- **portAddress** (Not used for OEM adapters)
- **memoryAddress** (Not used for OEM Adapters)

Return Value

The **OpenAdapter** function returns one of the following values:

```
0x00 = Adapter is opened successfully
0x01 = Adapter is already open
0x02 = Adapter not found
0x03 = IRQ already in use or out-of-range for this adapter
0x04 = Port Address already in use or out-of-range for this adapter
0x05 = Memory Address already in use or out-of-range for this adapter
0x06 = Error occurred while testing adapter RAM, check your Memory address range
0x07 = PCMCIA configuration changed, Reboot the machine
0x13 = Invalid adapter number
0x18 = Memory allocation error
```

Connection Type

The **OpenAdapter** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

Open Station

```
ULONG OpenStation (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG connectionType,
                    ULONG keyboardType
                                        );
```

The **OpenStation** function is used to open one of the stations on the adapter. When the **OpenStation** function is called, the MAC driver should check the parameters for the **OpenStation** function, the MAC driver's internal state, system resources to open a new station and so on. It should also verify that the address is not used by another device on the same port before opening the station. If an error is found during the opening process then a non-zero value should be returned to the upper

layer. If there is no error, the MAC should return zero to the common driver and start the opening process. Upon the completion of the opening process, the MAC driver needs to call the **OpenComplete** function to let the upper layer know the result of the process.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

- **stationAddress**

```
0x00 : Address 0
0x01 : Address 1
0x02 : Address 2
0x03 : Address 3
0x04 : Address 4
0x05 : Address 5
0x06 : Address 6
```

- **connectionType**

```
0x00 = APPC or SNA Pass-through (5150-001)
0x01 = SBCS Display (3197-C1)
0x02 = SBCS Printer (5224-001)
0x03 = SBCS Printer (5256-002)
0x04 = SBCS Printer (5219-D01)
0x05 = DBCS Display (5555-C01)
0x06 = DBCS Printer (5553-B01)
```

- **keyboardType** (SBCS/DBCS Display only)

  This parameter should include an extended keyboard ID which is sent to the workstation controller in the third byte of the data frames by "Activate Read" command following "Read Data" pre-activate command addressed to Model ID. For more information about the extended keyboard ID, please refer to the description of " Read Data Command" in the document titled "5250 Information Display System to System/36, System/38, and Application System/400 System Units Product Attachment Information".

Return Value

The **OpenStation** returns one of the following values:

```
0x00 = The process of opening station is started successfully
0x01 = No more address is available
0x02 = Specified address is already in use by another station
0x12 = Station address is invalid
0x14 = Adapter is not open
0x16 = Invalid connection type
0x18 = Error occurs when allocating memory for MAC Driver's object
```

Connection Type

The **OpenStation** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

Send Data

```
ULONG SendData (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    UCHAR *pSendData,
                    ULONG dataLength
                                        );
```

The **SendData** function is used to send data to the local/remote workstation controller. When the **SendData** function is called and there is no error on the parameters passed and the MAC driver's internal status, the MAC driver should prepare for sending the data and return 0x00. The upper layer driver waits for the **SendComplete** function call (confirmation). The buffer used for the **SendData** function is not deallocated until the MAC driver calls the **SendComplete** function or the **Timeout** function (when error occurs). Also, the common driver should not call any additional **SendData** function, for the same station address, until the MAC driver calls the **SendComplete** function for the outstanding **SendData** function. The MAC driver should return 0x15 if the common driver calls **SendData** prior to the **SendComplete**. The common driver does not divide a block of data into multiple data blocks. Therefore, when the **SendData** function is called, the MAC driver can immediately start preparing for sending of the data without waiting for the additional data from the common driver.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

- **pSendData** : Points to the user data buffer

- **dataLength** : Data length to be sent to the local/remote workstation controller

Return Value

The **SendData** function returns one of the following values:

```
0x00 = Data is sent successfully
0x02 = Station is not open
0x14 = Adapter is not open
0x15 = Internal Error (SendData outstanding)
```

Connection Type

The **SendData** function is used for the following connection types:

- APPC
- SNA Pass-through (3270 via AS/400)

Close Station

```
ULONG CloseStation (
                    ULONG adapterNumber,
                    ULONG stationAddress
                                        );
```

The **CloseStation** function causes the MAC driver to deactivate the specified station immediately. When the **CloseStation** function is called, the MAC driver should deallocate all resources which were allocated during the **OpenStation** function call, and the station should be made available for re-use.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

- **stationAddress**

```
0x00 : Address 0
0x01 : Address 1
0x02 : Address 2
0x03 : Address 3
0x04 : Address 4
0x05 : Address 5
0x06 : Address 6
```

Return Value

The **CloseStation** function returns one of the following values:

```
0x00 = Station is closed successfully
0x14 = Adapter is not open
```

Connection Type

The **CloseStation** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

Close Adapter

```
ULONG CloseAdapter (
                    ULONG adapterNumber
                                        );
```

The **CloseAdapter** function is used to close the adapter. If there is an active station when the **CloseAdapter** function is called, the MAC driver should not deallocate all resources allocated for the adapter, and should return 0x01. If there is no active station, all resources for the adapter number should be deallocated.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

Return Value

The **CloseAdapter** function returns one of the following values:

```
0x00 = Adapter is closed successfully
0x01 = One or more sessions are still active
0x14 = Adapter is not open
```

Connection Type

The **CloseAdapter** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

Send Scan Code

```
ULONG SendScanCode (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG scanCode
                                        );
```

The **SendScanCode** function is used to send the workstation controller a scan code from SBCS/DBCS display emulation program. When the **SendScanCode** function is called, the MAC driver checks to see if the station is in online state or not. If it is in online state, a scan code in the **scanCode** parameter is queued in the MAC driver's scan code queue. If it is not in online state, the **SendScanCode** function does nothing. When this function is called while the MAC driver's scan code queue reaches the maximum count, the MAC driver replaces the last position in the queue with the overrun scan code (0xFF) so that the workstation controller sends error code "0001" to the left corner of the 24th line on the screen.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

- **scanCode**

  This parameter should include a scan code which is sent to the workstation controller.

Return Value

The **SendScanCode** function returns one of the following values:

```
0x00 = Scan Code is queued in the MAC driver's queue
0x15 = Internal Error
```

Connection Type

The **SendScanCode** function is used for the following connection types:

- SBCS Display
- DBCS Display

Send Printer Status

```
ULONG SendPrtStatus (
                        ULONG adapterNumber,
                        ULONG stationAddress,
                        ULONG RspStatus1,
                        ULONG RspStatus2,
                        ULONG PrtStatus
                                              );
```

The **SendPrtStatus** function is used to send the printer status to the workstation controller. According to the status bits set in the **RspStatus1** and **RspStatus2** parameter, the MAC driver updates the poll responses. Once some status bits except "UNA" and "EOF" bits in the **RspStatus1** or **RspStatus2** parameter are set by the MAC driver after this function is called, it is automatically reset at a appropriate timing. As for the "UNA" bit and "EOF" bit, this function should called again. In this case, the status bit(s) to be reset and Set/Reset request indicator in the **RspStatus1** or **RspStatus2** parameter should be set to ON. The **PrtStatus** parameter is used only when the outstanding status bit in the **RspStatus1** parameter is set to ON. This status byte will be sent to the workstation controller by "Activate Read" following "Read Status" pre-activate command. This function is not used to update "Print Complete" and "Receive Buffer Full" status in the poll response. These bits are controlled by the MAC driver based on the buffer availability.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

- **RspStatus1**

```
7  6  5  4  3  2  1  0
-  -  -  -  -  -  -  -
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  Not used
|  |  |  |  |  |  Not used
|  |  |  |  |  Set/Reset Request Indicator
|  |  |  |  |   0 : Set Request
|  |  |  |  |   1 : Reset Request
|  |  |  |  Not used
|  |  |  Outstanding Status
|  |  Unit Not Available
|  Not used
Not used
```

- **RspStatus2**

```
7  6  5  4  3  2  1  0
-  -  -  -  -  -  -  -
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  Graphic Check (unprintable character)
|  |  |  |  |  |  End of Form
|  |  |  |  |  Set/Reset Request Indicator
|  |  |  |  |   0 : Set Request
|  |  |  |  |   1 : Reset Request
|  |  |  |  Cancel
|  |  |  Not used
|  |  Not used
|  Invalid SCS Parameter
Invalid SCS Control Code
```

- **PrtStatus**

  The printer hardware error is set in this parameter. The values are different for different printers. Please refer to the printer's technical information for the meaning for each printer. Types of errors set in the parameter are Machine Checks, Data Stream Exceptions, etc.

Return Value

The **SendPrinterStatus** function returns one of the following values:

```
0x00 = The MAC driver receives the new printer status
0x15 = Internal Error
```

Connection Type

The **SendPrinterStatus** function is used for the following connection types:

- SBCS Printer
- DBCS Printer

Print Complete

```
ULONG PrintComplete (
                        ULONG adapterNumber,
                        ULONG stationAddress
                                              );
```

The **PrintComplete** function is used to inform the MAC driver that the upper layer (Printer emulation program) has finished processing the SCS data which is filled in one of the two printer receive buffers. When this function is called, the MAC driver updates the poll response frame 2 in accordance with the current poll response status. That is to say, if the adapter is currently sending the "Print Complete" OFF status and the "Receive Buffer Full" ON status, the "Receive Buffer Full" bit is set to OFF. And if the adapter is currently sending the "Print Complete" OFF status and the "Receive Buffer Full" OFF status, the "Print Complete" bit is set to ON.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

Return Value

The **PrintComplte** function returns one of the following values:

```
0x00 = The PrintComplte function is processed successfully
0x15 = Internal Error
```

Connection Type

The **PrintComplete** function is used for the following connection types:

- SBCS Printer
- DBCS Printer

Clear Complete

```
ULONG ClearComplete (
                        ULONG adapterNumber,
                        ULONG stationAddress
                                              );
```

The **ClearComplete** function is used to inform the MAC driver that the upper layer (Printer emulation program) has finished processing the CLEAR command process. When this function is called, the "Receive Buffer Full" bit must be set to OFF and the "Print Complete" bit must be set to ON.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

Return Value

The **ClearComplte** function returns one of the following values:

```
0x00 = The ClearComplte function is processed successfully
0x15 = Internal Error
```

Connection Type

The **ClearComplete** function is used for the following connection types:

- SBCS Printer
- DBCS Printer

3.3.2 MAC library functions

The MAC driver can call the functions defined in the MAC library. The functions in the MAC library provide a convenient way to call the upper layer directly based on the function table and to set the function table when an IRP or an IOCTL is received from the upper layer driver. All functions except **SetFunctionTables** function should be called at the dispatch level. The **SetFunctionTables** function should be called at the passive or user level.

Get Buffer

```
UCHAR *GetBuffer (
                    ULONG adapterNumber,
                    ULONG stationAddress
                    ULONG length
                                        );
```

The **GetBuffer** function is used to get a buffer from the upper layer for the incoming data. The upper layer allocates the buffer from non-paged pool and locks it before returning from this function.

If the connection type is APPC or SNA Pass-through, the buffer is used to copy the incoming data from the MAC driver's DMA buffer and to send it to the upper layer. The MAC driver needs to get a new buffer each time it receives one data block, which always includes the TDLC header and the whole data of the length which is set in the first two bytes of the TDLC header. (One data block may come by one or more **Activate Write** commands.) The MAC driver can assume that the buffer size allocated by the **GetBuffer** function is enough to receive the whole incoming data block (The buffer size should always be greater than or equal to (max frame size + 64) the length which is set in the first two bytes of TDLC header.) However, the MAC driver needs to set the required length in the **length** parameter to let the upper layer check to see if a memory overflow will occur. The MAC driver should set the value based on the information from TDLC header. If the buffer length set in the **length** parameter is bigger than the size the upper layer supports, an error message is logged and a NULL pointer is returned. The buffer which is obtained by the **GetBuffer** function is automatically unlocked and deallocated when it is processed by the upper layer.

If the connection type is DBCS/SBCS display, the buffer is used to save the screen data. The MAC driver needs to call this function to allocate the screen size buffer once when the **OpenStation** function is called. And after the process of the **OpenStation** function completes, it is used as a shared buffer between the MAC driver and the upper layer until the station is closed. The buffer is automatically deallocated by upper layer after the **CloseStation** function is called. The MAC driver needs to update the data in this buffer whenever the workstation controller sends data to PC so that the upper layer knows the latest screen data.

If the connection type is DBCS/SBCS printer, the buffer is used to receive SCS data. The MAC driver needs to call this function to get a new buffer each time the MAC driver receives the SCS data. The MAC driver can assume that the buffer size allocated by the **GetBuffer** function is enough to receive the whole incoming

SCS data (which should be equal to or less than 256). The buffer is automatically freed by the upper layer after the **PrintData** function is called.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

Return Value

The **GetBuffer** function returns one of the following values:

```
Buffer Address (when buffer is available)
NULL (when buffer is not available)
```

Connection Type

The **GetBuffer** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

Open Complete

```
ULONG OpenComplete (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG errorCode
                                        );
```

The **OpenComplete** function is used to inform the upper layer that the Open Station process has completed. The result of the open station process is set into **errorCode** field. An error can occur when the station address is already used by another device or the Twinax cable is disconnected.

Note: Even if the MAC driver does not support the line checking function before opening the adapter, the **OpenComplete** function must be called to let the upper

layer know that the MAC driver has completed the opening process. In this case, the user needs to be careful not to use a station address which is already used by another device on the same port. This line monitoring function should not be done for display connection type so that it can be used for an AS/400 console.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

- **stationAddress**

```
0x00 : Address 0
0x01 : Address 1
0x02 : Address 2
0x03 : Address 3
0x04 : Address 4
0x05 : Address 5
0x06 : Address 6
```

- **errorCode**

```
0x00 = The Open Station process has completed successfully.
0x02 = Address is in use by another device on the same port.
0x04 = There is no polling from remote/local controller or
       the cable is disconnected.
0x05 = Line parity error is detected during opening process.
```

Return Value

The **OpenComplete** function returns one of the following values:

```
0x00 = The OpenComplete function is processed successfully
0x15 = An internal error occurred in the process of OpenComplete function
```

Connection Type

The **OpenComplete** function is used for the following connection types:

- APPC
- SNA Pass-through (3270 via AS/400)

Receive Data

```
ULONG ReceiveData (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG dataLength
                                        );
```

The **ReceiveData** function is used to pass the incoming data to the upper layer. All data from the local/remote workstation controller must be copied in the buffer which is obtained by a **GetBuffer** function. The upper layer driver assumes, when the **ReceiveData** function is called, that the buffer address is the same as the one returned to the MAC driver in the preceding **GetBuffer** function. The MAC driver needs to pass all data from the local/remote workstation controller without modification except data sent to SS Message Buffer (Arctic Address: 0xE000) or Read Response Field (Arctic Address: 0xF000), which should be handled in the MAC driver. When one data block is divided into multiple pieces, which requires the multiple **Activate Write** command, the MAC driver needs to wait for data arrival before calling the **ReceiveData** function until all data has been received. To check how many bytes the local/remote workstation controller will send for one data block, the MAC driver needs to check the first two bytes (which are written into the Arctic address 0x9000-0x9001) and wait for a while until all data arrive.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

- **dataLength** : Length of the data which the local/remote workstation controller sends

Return Value

The **ReceiveData** function returns one of the following values:

```
0x00 = The ReceiveData function is processed successfully
0x15 = An internal error occurred in the process of ReceiveData function
```

Connection Type

The **ReceiveData** function is used for the following connection types:

- APPC
- SNA Pass-through (3270 via AS/400)

Send Complete

```
ULONG SendComplete (
                    ULONG adapterNumber,
                    ULONG stationAddress
                                        );
```

The **SendComplete** function is used to confirm the upper layer that the data in the previous **SendData** function is successfully sent to the local/remote workstation controller. When this function is called, the upper layer driver will deallocate the buffer used for the **SendData** function. The best time to call the **SendComplete** function is when the local/remote workstation controller sends a **Write Data Load Cursor** commnad to the status line device after the outgoing data is sent by **Activate Read** command(s).

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

Return Value

The **SendComplete** function returns one of the following values:

```
0x00 = The SendComplete function is processed successfully
0x15 = An internal error occurred in the process of SendComplete function
```

Connection Type

The **SendComplete** function is used for the following connection types:

- APPC
- SNA Pass-through (3270 via AS/400)

Timeout

```
ULONG Timeout (
                ULONG adapterNumber,
                ULONG stationAddress
                                    );
```

The **Timeout** function is used to inform the upper layer that the local/remote workstation controller did not send a poll in last 2 seconds or more, or Twinax cable is disconnected.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

Return Value

The **Timeout** function returns one of the following values:

```
0x00 = The Timeout function is processed successfully
0x15 = An internal error occurred in the process of Timeout function
```

Connection Type

The **Timeout** function is used for the following connection types:

- APPC
- SNA Pass-through (3270 via AS/400)

Load Cursor Position

```
ULONG LoadCursor (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG cursorPosition
                                        );
```

The **LoadCursor** function is used to inform the upper layer of new cursor position. This function is called when the workstation controller sends some Twinax commands which requires cursor movement, the initialization completes or the line is disconnected.

Parameters

- **adapterNumber**

  ```
  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2
  ```

- **stationAddress**

  ```
  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6
  ```

- **cursorPosition**

  ```
  0x0000 - 0x07FF : Offset to Row 1, Column 1

   0xYYYY ZZZZ
     ---- ----
      |    |
      |    +----  Regular Cursor Position
      +---------  Shadow Cursor Position

                  0x0000 - 0x07FF : Offset to Row 1, Column 1
  ```

Return Value

The **LoadCursor** function returns one of the following values:

```
0x00 = The LoadCursor function is processed successfully
0x15 = An internal error occurred in the process of LoadCursor function
```

Connection Type

The **LoadCursor** function is used for the following connection types:

- SBCS Display
- DBCS Display

Display Function Control

```
ULONG DisplayControl (
                        ULONG adapterNumber,
                        ULONG stationAddress,
                        ULONG controlData
                                              );
```

The **DisplayControl** function is used to pass the control data in the data frames following "Write Control Data" command. The MAC driver will process some of the bits on the control data such as "reset exception status" bit and pass all control data in the data frames so that the upper layer (the display emulation program) can process the display control function.

Note: Some of the bits defined in the control data following "Write Control Data" is not currently supported. For instance, the setting which the workstation controller specified such as "cursor blink control or background control" are not processed. Instead of that, the setting locally configured is used.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

- **stationAddress**

```
0x00 : Address 0
0x01 : Address 1
0x02 : Address 2
0x03 : Address 3
0x04 : Address 4
0x05 : Address 5
0x06 : Address 6
```

- **controlData**

```
0x0000xxxx
      ----
      | |
      | +---  First Control Data
      +-----  Second Control Data (if any)
```

  This parameter should include the control data sent from the workstation controller. For more information about the definition of he control data, please refer to the description of "Write Control Data Command" in the document titled "5250 Information Display System to System/36, System/38, and Application System/400 System Units Product Attachment Information".

Return Value

The **DisplayControl** function returns one of the following values:

0x00 = The **DisplayControl** function is processed successfully
0x15 = An internal error occurred in the process of **DisplayControl** function

Connection Type

The **DisplayControl** function is used for the following connection types:

- SBCS Display
- DBCS Display

Display Mode

```
ULONG DisplayMode (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG displayMode
                                        );
```

The **DisplayMode** function is used to change the display mode which is set by "Set Mode" command. If the bits 13-14 in the data frame of "Set Mode" command is 0b00, the display goes into normal mode. If those are 0b11, the display goes into display off mode.

Parameters

- **adapterNumber**

  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2

- **stationAddress**

  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6

- **displayMode**

  0x00 : Normal Mode
  Not 0 : Display Off Mode

Return Value

The **DisplayMode** function returns one of the following values:

0x00 = The **DisplayMode** function is processed successfully
0x15 = An internal error occurred in the process of **DisplayMode** function

Connection Type

The **DisplayMode** function is used for the following connection types:

- SBCS Display
- DBCS Display

Screen Update

```
ULONG ScreenUpdate (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG updatePosition,
                    ULONG updateLength
                                        );
```

The **ScreenUpdate** function is used to inform the upper layer (display emulation program) that the presentation space has been updated. The screen data in the buffer allocated by calling **GetBuffer** function must be updated before calling this function.

Parameters

- **adapterNumber**

  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2

- **stationAddress**

  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6

- **updatePosition**

  0x0000 - 0x07FF : Offset to Row 1, Column 1

- **updateLength**

  0x0001 - 0x0800

Return Value

The **ScreenUpdate** function returns one of the following values:

0x00 = The **ScreenUpdate** function is processed successfully
0x15 = An internal error occurred in the process of **ScreenUpdate** function

Connection Type

The **ScreenUpdate** function is used for the following connection types:

- SBCS Display
- DBCS Display

OIA Update

```
ULONG OIAUpdate (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG oiaIndicator,
                    ULONG saIndicator
                                        );
```

The **OIAUpdate** function is used to inform the upper layer (display or printer emulation program) that the Operator Information Area has been updated. When the connection type is display, the MAC driver sends the System Available Indicator and the data frame following "Write Data Load Cursor" command addressed to the indicator device. When the connection type is printer, only **saIndicator** is used to let an end user know the status of the connection.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

- **stationAddress**

```
0x00 : Address 0
0x01 : Address 1
0x02 : Address 2
0x03 : Address 3
0x04 : Address 4
0x05 : Address 5
0x06 : Address 6
```

- **oiaIndicator**

```
xx = The bits definition is the same as that of data frame following "WDLC" command

       7  6  5  4  3  2  1  0
       -  -  -  -  -  -  -  -
       |  |  |  |  |  |  |  |
       |  |  |  |  |  |  |  Not used
       |  |  |  |  |  |  Input Inhibit
       |  |  |  |  |  Not used (Diacritic Indicator)
       |  |  |  |  Insert Mode
       |  |  |  Not used (Katakana shift)
       |  |  Upper shift
       |  Not used
       Message Waiting
```

- **saIndicator**

```
0x00  : System Available Indicator OFF
Not 0 : System Available Indicator ON
```

Return Value

The **OIAUpdate** function returns one of the following values:

```
0x00 = The OIAUpdate function is processed successfully
0x15 = An internal error occurred in the process of OIAUpdate function
```

Connection Type

The **OIAUpdate** function is used for the following connection types:

- SBCS Display
- DBCS Display
- SBCS Printer
- DBCS Printer

Reset Printer

```
ULONG ResetPrinter (
                    ULONG adapterNumber,
                    ULONG stationAddress
                                        );
```

The **ResetPrinter** function is used to inform the upper layer (printer emulation program) that the Twinax "Reset" command is received. After this function is called, the upper layer will close and re-open the station.

Parameters

- **adapterNumber**

```
0x00 = Adapter 0
0x01 = Adapter 1
0x02 = Adapter 2
```

- **stationAddress**

```
0x00 : Address 0
0x01 : Address 1
0x02 : Address 2
0x03 : Address 3
0x04 : Address 4
0x05 : Address 5
0x06 : Address 6
```

Return Value

The **ResetPrinter** function returns one of the following values:

0x00 = The **ResetPrinter** function is processed successfully
0x15 = An internal error occurred in the process of **ResetPrinter** function

Connection Type

The **ResetPrinter** function is used for the following connection types:

- SBCS Printer
- DBCS Printer

Clear Printer Buffer

```
ULONG ClearBuffer (
                    ULONG adapterNumber,
                    ULONG stationAddress
                                        );
```

The **ClearBuffer** function is used to inform the upper layer (printer emulation program) that the Twinax "Clear" command is received. When "Clear" command is received, the MAC driver sets the "Receive Buffer Full" bit to ON and calls this function and the MAC driver continues to send "Receive Buffer Full" bit until the upper layer calls **ClearComp** function. When **ClearComp** function is called, the MAC driver reset "Receive Buffer Full" and set "Print Complete" bit to ON.

Parameters

- **adapterNumber**

  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2

- **stationAddress**

  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6

Return Value

The **ClearBuffer** function returns one of the following values:

0x00 = The **ClearBuffer** function is processed successfully
0x15 = An internal error occurred in the process of **ClearBuffer** function

Connection Type

The **ClearBuffer** function is used for the following connection types:

- SBCS Printer
- DBCS Printer

Print Data

```
ULONG PrintData (
                    ULONG adapterNumber,
                    ULONG stationAddress,
                    ULONG dataLength
                                        );
```

The **PrintData** function is used to inform the upper layer (printer emulation program) that the SCS data arrived. When the MAC driver receives the SCS data, the "Receive Buffer Full" bit and "Print Complete" bit are updated based on the current buffer availability. That is to say, when "Print Complete" bit was set to ON and "Receive Buffer Full" was set to OFF, "Print Complete" bit will be set to OFF. When "Print Complete" bit was set to OFF and "Receive Buffer Full" bit was set to OFF, "Receive Buffer Full" will be set to ON. The upper layer assumes, when the **PrintData** function is called, that the buffer address is the same that returned to the MAC driver in the preceding **GetBuffer** function.

Parameters

- **adapterNumber**

  0x00 = Adapter 0
  0x01 = Adapter 1
  0x02 = Adapter 2

- **stationAddress**

  0x00 : Address 0
  0x01 : Address 1
  0x02 : Address 2
  0x03 : Address 3
  0x04 : Address 4
  0x05 : Address 5
  0x06 : Address 6

- **dataLength**

  0x0001 - 0x0100

Return Value

The **PrintData** function returns one of the following values:

0x00 = The **PrintData** function is processed successfully
0x15 = An internal error occurred in the process of **PrintData** function

Connection Type

The **PrintData** function is used for the following connection types:

- SBCS Printer
- DBCS Printer

Set Function Tables for Windows NT

```
NTSTATUS SetFunctionTables (
                              PDEVICE_OBJECT device,
                              PIRP           irp
                                                  );
```

The **SetFunctionTables** function is used to set up the function table which is used for the direct calls to the upper layer driver. The MAC driver should call this function in the dispatch routine for **IoDeviceControl** (IRP_MJ_DEVICE_CONTROL). If the IRP comes from the upper layer driver when the MAC driver's dispatch routine is called, the **SetFunctionTables** function returns STATUS_SUCCESS (defined in the DDK's include file). Otherwise, it returns STATUS_NOT_IMPLEMENT.

Parameters

- **device** : Points to the device object
- **irp** : Points to the IRP

Return Value

The **SetFunctionTables** function returns one of the following values:

```
STATUS_SUCCESS (0x00000000) : The function table is set up successfully
STATUS_NOT_IMPLEMENTED (0xC0000002) : The sender of the IRP is not common driver
```

Connection Type

The **SetFunctionTables** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

Set Function Tables for Windows 95

```
ULONG SetFunctionTables (
                          void      *pInBuffer,
                          ULONG     lInBuffer,
                          void      *pOutBuffer,
                          ULONG     *plOutBuffer
                                                  );
```

The **SetFunctionTables** function is used to set up the function table which is used for the direct calls to the upper layer driver. The MAC driver should call this function in the W32DeviceIoControl routine when it receives **eSetFunctionTables** I/O control code. This function always returns 0.

Parameters

- **pInBuffer** : Address of the IOCTL's input buffer
- **lInBuffer** : Length of the input buffer
- **pOutBuffer** : Address of the IOCTL's output buffer
- **plOutBuffer** : Address of the IOCTL's output buffer length

Return Value

The **SetFunctionTables** function set the length of the output buffer in *plOutBuffer*, and returns 0.

Connection Type

The **SetFunctionTables** function is used for the following connection types:

- APPC
- SBCS Display
- SBCS Printer
- DBCS Display
- DBCS Printer
- SNA Pass-through (3270 via AS/400)

4.0 Requirements

4.1 Windows NT

The OEM MAC driver's internal device name must be "TwinaxOEM".

A service key for the OEM MAC driver should be created in the Registry (see HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\pdlnatsn (PComm/CSNT) or ibmtwxsn (CA/400) IBM Twinax MAC driver service key). The OEM MAC driver service key should be added to the DependOnService key of HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\pdlnatcm (PComm or CSNT) or ibmtwx (CA/400) service.

For IBM compatible PCMCIA adapters, the install program of the OEM adapter should add the HardwareID to PCMCIA data base in the Registry (HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\Pcmcia\DataBase\xyz co.), and create a string key "Driver" with value "pdlnatsn" (PComm and CSNT) or "ibmtwxsn" (CA/400).

Note: APAR ICXXXXX is required (from IBM) for Personal Communications Version 4.1 for Windows NT, and APAR ICXXXXX is required for IBM Communications Server for Windows NT Version 5.01.

4.2 Windows 95

The install program of the OEM adapter should create or update the following registry keys:

- HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Twinax\OldOEMVxD

  This is a DWORD value and it must be set to zero. If the key does not exist or it is set to a non-zero value, then a Windows 3.1 OEM driver will be used.

- HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Twinax\DeviceName

  This is a String value and it must be set to the OEM device driver's file name (8.3 format) without the path (the file extension must be .VxD). The full directory path of the VxD file must be in the PATH environment variable.

- HKEY_LOCAL_MACHINE\SOFTWARE\IBM\Twinax\HardwareID

  This is a String value and it must be set to the OEM adapter's Tuple HardwareID. This value is used to search the Registry for PnP (ISA PnP, PCI and PCMCIA) adapter/card.

  An .inf file should be provided on the adapter diskette.

End of Document