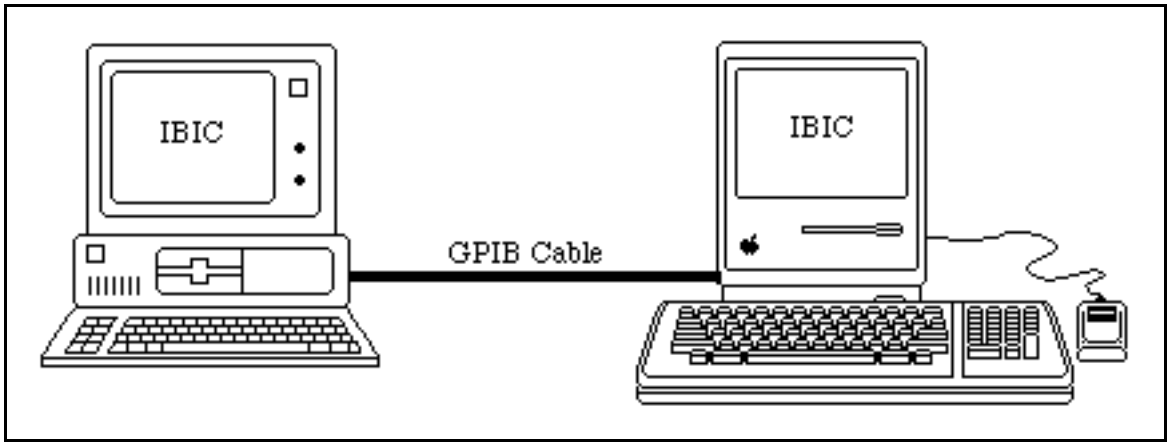
Interactive File Transfers Using the GPIB

T. Dehne and D. Scheibenhoffer

Introduction

This application note explains how to transfer data from one computer to another computer using National Instruments IEEE 488 interface boards for the PC and the Interface Bus Interactive Control (`IBIC`) Program.



`IBIC` is part of the National Instruments IEEE 488 software (NI-488®) that is included with your GPIB interface board. The NI-488 software package consists of a high-speed device driver that is usually installed as part of the operating system and several utilities that help develop and debug an application program. The software is designed so that the user does not have to learn the programming details of the GPIB interface board or the IEEE 488 protocol. Low-level functionality, however, is also available for maximum flexibility and performance.

The first generation of the NI-488 driver, defined by the ANSI/IEEE Standard 488.1-1987, was the first loadable IEEE 488 device driver for MS-DOS-based personal computers. NI-488 drivers are now available for a number of different computer platforms, including the IBM PC and PS/2, Macintosh, Sun, DEC, HP, Apollo, and NEC, and a number of different operating systems such as DOS, Windows 3, OS/2, UNIX, Solaris, AIX, ULTRIX, XENIX, and 386/ix. NI-488 programs are portable across these different platforms. The new generation of NI-488 software, named *NI-488.2™, includes a set of NI-488.2 routines that let you take full advantage of the ANSI/IEEE Standard 488.2-1987 capabilities.

---

* Product and company names listed are trademarks or trade names of their respective manufacturers.

This application note details a data transfer between two PCs; however, the method can be used to transfer data between any of the computers or operating systems listed in the previous paragraph.

Applications

The GPIB is a standard communication link that operates independently of the computer or operating system. A number of applications can be simplified by using the GPIB to transfer data from one computer to another. Consider the following examples:

- Transferring data between a Macintosh and a Sun workstation, or between a PC running AT&T UNIX and a PC running MS-DOS can be a tedious task, usually involving a network or modems.

- A PC is monitoring an experiment and collecting data in the laboratory. When the application is complete, you must transfer the data to another computer to perform analysis, plotting/graphing, or spreadsheet and database manipulation. This type of floppy disk transfer can be time-consuming and frustrating.

- Several computers are being used within a laboratory or work area. These computers must communicate with each other so that information can be shared and messages transferred.

Using the GPIB to transfer data between computers is simple and inexpensive. The `IBIC` program, which is included with the NI-488 software, lets you transfer data interactively from the keyboard without having to write a program.

Configuring the Computers

A typical GPIB system includes a Controller and a number of other devices such as oscilloscopes, multimeters, logic analyzers, printers, and plotters. The computer is usually the *System Controller* and has full control of these devices. In a computer-to-computer data transfer setup, however, one of the computers must be the System Controller and the other must act as a device or *Non-System Controller*. In addition, each computer must be configured for a different GPIB primary address. The GPIB configuration program (`IBCONF`), supplied with the NI-488 software, easily configures each computer for its respective role in the transfer.

To configure the computers for data transfer, follow these steps:

1. Decide which computer will be System Controller and which will be Non-System Controller. The assignment is arbitrary.

2. Install the NI-488 software and test the software installation.

3. Run the `IBCONF` program on each computer.

   Note: If you did not change the default settings of the System Controller, there is no need to run `IBCONF`.

a. Start `IBCONF` by entering:

```
c:\> ibconf      <Enter>
```

b. Edit the `gpib0` board characteristics by pressing the <F8> function key.

c. Edit the characteristics according to Table 1.

Note: Table 1 shows the board characteristics for two AT-GPIB boards. The characteristics are similar for other National Instruments GPIB boards. NI-488.2 software for Macintosh is configurable through the Control Panel.

Table 1. Characteristics of the AT-GPIB

System Controller		**Non-System Controller**	
Primary GPIB Address	**0**	**Primary GPIB Address**	**1**
Secondary GPIB Address	NONE	Secondary GPIB Address	NONE
Timeout setting	T10s	Timeout setting	T10s
EOS byte	00H	EOS byte	00H
Terminate Read on EOS	no	Terminate Read on EOS	no
Set EOI with EOS on Write	no	Set EOI with EOS on Write	no
Type of compare on EOS	7-bit	Type of compare on EOS	7-bit
Set EOI w/last byte of Write	yes	Set EOI w/last byte of Write	yes
Board is System Controller	**yes**	**Board is System Controller**	**no**
Assert REN when SC	no	Assert REN when SC	no
Enable Auto Serial Polling	yes	Enable Auto Serial Polling	yes
Enable NI-488 protocols	yes	Enable NI-488 protocols	yes
CIC protocols	yes	CIC protocols	yes
Timing	500nsec	Timing	500nsec
Interrupt jumper setting	11	Interrupt jumper setting	11
Base I/O Address	02C0H	Base I/O Address	02C0H
DMA channel	5	DMA channel	5

For the GPIB board designated as System Controller, the *`System Controller`* field is set to *`yes`* and the field *`Primary GPIB Address`* set to *`0`*. These are the default settings for the NI-488 driver. If these settings have not been changed, there is no need to run `IBCONF` for the System Controller. The Non-System Controller is configured with the *`System Controller`* field set to *`no`* and the field *`Primary GPIB Address`* set to *`1`*. `IBCONF` must be run for the Non-System Controller to change these settings.

d. Exit `IBCONF`. Save your changes to the loaded driver in memory, if prompted.

Note: When you exit `IBCONF` a caution message appears on the Non-System Controller computer screen to alert you that both `gpib0` and `dev1` are configured for the same GPIB primary address. This addressing conflict is a problem only if another device is actually connected to the GPIB with a primary address of 1 and it is powered on. If there is no such device, the caution message can be ignored by typing a *`y`* at the prompt *`Do you still wish to exit IBCONF?`*

(y/n). If you made any changes, simply save your changes to the loaded driver in memory.

Establishing Communication

The IBIC program lets you use all the NI-488 functions and subroutines from the keyboard. This is a quick and convenient method for establishing communication with GPIB devices without having to write an entire application program. This is an excellent tool for debugging a GPIB command sequence, assisting development of an application program, verifying communication with a device, or even transferring files from PC to PC.

Execute the IBIC program on *both* computers by first moving to the directory that contains all the NI-488 files as shown in the following example. (The name of the directory that holds the NI-488 files depends on which board you have.) For this example, data will be transferred between two AT-GPIB boards.

Note: IBIC prompts are in boldface in all the examples in this application note. User inputs are italicized and must be followed by pressing the <Enter> key.

```
c:\>  cd at-gpib
```

Next, enter *ibic* at the prompt as shown in the following example:

```
c:\at-gpib>  ibic
```

After you have executed the IBIC program, enter the commands as shown in the following examples:

System Controller

```
: ibfind dev1
```

Non-System Controller

```
: ibfind gpib0
```

Each computer responds as follows:

System Controller

```
dev1:
```

Non-System Controller

```
gpib0:
```

The System Controller accesses the device `dev1` because `dev1` is at primary address 1 by default. This is the address previously assigned to the Non-System Controller. Because the System Controller opened `dev1`, it uses high-level (device) functions to communicate with `dev1`. The high-level NI-488 functions automatically take care of the bus management details, such as asserting Interface Clear (IFC) and addressing devices for proper communication. The Non-System Controller must communicate using the low-level (board) functions.

Transferring Data

To transfer data, one system must write data while the other system reads data. To transfer data from the System Controller to the Non-System Controller, type the commands in the following examples and press <Enter> after each command.

System Controller

```
dev1: ibwrt "HELLO"
```

Non-System Controller

```
gpib0: ibrd 5
```

The System Controller sends the data string to the Non-System Controller, and each responds as follows:

System Controller

```
dev1: ibwrt "HELLO"
[0100] (cmpl)
count: 5
```

Non-System Controller

```
gpib0: ibrd 5
[21C4] (end cmpl lok rem lacs)
count: 5
48 45 4C 4C 4F             HELLO
```

Note: The read and write commands must be executed within a certain time limit of each other; otherwise the timeout setting specified in `IBCONF` (default = 10 sec) aborts the operation.

Data can also be passed from the Non-System Controller to the System Controller by reversing the write and read commands, as shown in the following example:

System Controller

```
dev1: ibrd 5
```

Non-System Controller

```
gpib0: ibwrt "HELLO"
```

The Non-System Controller sends the data string to the System Controller, and each responds as follows:

System Controller

```
dev1: ibrd 5
[2100] (end cmpl)
count: 5
48 45 4C 4C 4F       HELLO
```

Non-System Controller

```
gpib0: ibwrt "HELLO"
[0188]  (cmpl lok tacs)
count: 5
```

Transferring Files

System Controller to Non-System Controller

Choose a file from the System Controller for transfer. If a large data file is chosen, make sure the timeout settings in `IBCONF` are increased to avoid I/O termination before the file transfer is complete.

Replace *filename* with the drive designator, pathname, and filename for the source file and destination file. If the default drive and pathname are being used, only the filename is required.

System Controller

```
: ibfind dev1
dev1: ibwrtf filename
```

Non-System Controller

```
: ibfind gpib0
gpib0: ibrdf filename
```

Note: Do not place the *filename* within quotation marks.

The System Controller transfers the file to the Non-System Controller, and each Controller responds with a message similar to the following:

System Controller

```
dev1: ibwrtf filename
[0100]  (cmpl)
count:  XXXXX
```

Non-System Controller

```
gpib0: ibrdf filename
[2144]  (end cmpl lok lacs)
count:  XXXXX
```

XXXXX is the number of bytes transferred. Make sure that count **XXXXX** is the same size as the original file. Although the filename used in the `ibwrtf` argument must exist on that particular computer system, the filename used in `ibrdf` can be designated as a new file. In other words, if that particular file does not exist, `ibrdf` creates a new file.

Note: The read and write commands must be executed within a certain time limit of each other; otherwise, the timeout setting specified in `IBCONF` (default = 10 sec) will abort the operation. If the file to be transferred is large, lengthen the timeout value to be certain that the whole file will be transferred.

Non-System Controller to System Controller

Data files can also be transferred from the Non-System Controller to the System Controller by reversing the read and write commands, as in the following examples:

System Controller

```
: ibfind dev1
dev1: ibrdf filename
```

Non-System Controller

```
: ibfind gpib0
gpib0: ibwrtf filename
```

The Non-System Controller sends the data file to the System Controller, and each responds as follows:

System Controller

```
dev1: ibrdf filename
[2124]  (end cmpl)
count:  XXXXX
```

Non-System Controller

```
gpib0:  ibwrtf filename
[0148]  (cmpl lok lacs)
count:  XXXXX
```

XXXXX is the number of bytes transferred. Make sure that count **XXXXX** is the same size as the original file.

Summary

The `IBIC` program is a quick and easy way to transfer data between computers. These `IBIC` examples can be used as the foundation for developing a complete application program. Such an application program can customize these transfers to your particular application. The same NI-488 functions (with some minor syntax differences) are used in the application program.