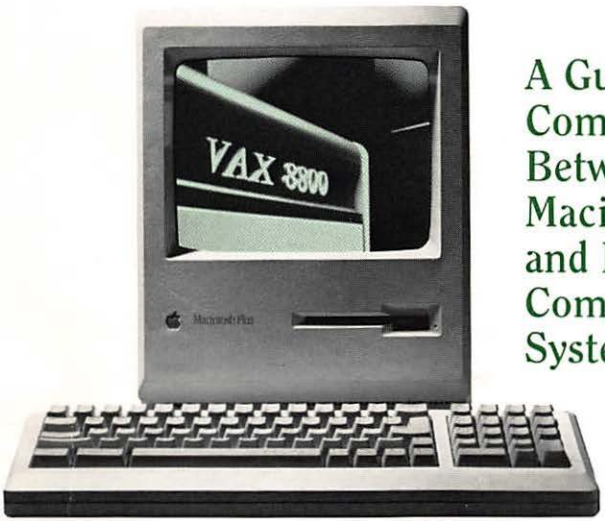
FILE TRANSFER



A Guide To Communications Between Apple Macintosh and DEC VAX Computer Systems.

WHITE PINE SOFTWARE

94 Route 101A, P.O. Box 1108, Amherst, NH 03031 (603) 886-9050

ABOUT THIS BOOKLET

White Pine Software, pioneers in Macintosh to VAX communications, has developed this booklet to take some of the mystery out of moving information between your Macintosh and VAX computer systems.

The intent of this booklet is to illustrate some of the uses and possibilities of file transfers. It touches upon the major topics involved in transferring files between the Macintosh and VAX systems. If you are a communications engineer, or are developing Macintosh communications software, you probably won't find anything new here. However, if transferring information between the Macintosh and VAX is new to you, or if you find it somewhat confusing, then you may benefit from this booklet.

The need to transfer and share information between different computer systems has brought about the development of many useful tools designed specifically for that purpose. A familiarity with computer communications concepts and the proper use of these tools will help you work more productively, and will enable you to take advantage of the capabilities unique to your various computer systems. Although we do use White Pine Software products in many of the examples, the concepts illustrated here will also apply to other vendor software packages, as well as other host and personal computer systems.

TABLE OF CONTENTS

TABLE OF CONTENTS

FILE TRANSFER CONCEPT

So your boss is letting you work at home today. You've got a Macintosh, and you're just putting the finishing touches on the ten page report he needs in a few days. The phone rings... it's your boss. There's been a change in schedule. He needs a hard copy of the report immediately. He also wants you to make it available to all the Macintosh and VAX users throughout the company. On top of that, he wants you to distribute a VAX mail message informing the department heads of the new report. But your car's in the shop, and you have no way of getting in to work. You don't even have access to a VAX terminal. You're stuck... but you don't have to be.

With your Mac, a modem, and the appropriate file transfer software, you could easily get yourself out of this mess. It's easy to dial-in to your VAX with a terminal emulator to create and mail a message to the department heads. While you're at it, you can quickly upload your report, unaltered, to a VAX based file storage application. Other Macintosh users can then copy the file for use on their Macs. You can also convert the file to a text format that the VAX users can read with their editor. And the printed report... no problem. Now that there's a VAX version of your report, just send it to the VAX printer. Finally, send your boss a VAX mail message telling him just where to go... (to get the report)!

This example illustrates some of the common uses for file transfers between computers. Recent developments in telecommunications and network technology have brought together computer systems from many manufacturers. In fact, the average user no longer is tied to any single computer. Users are beginning to take advantage of the strengths of a particular system for one task while using different systems to handle others.

Macintosh users can take some of the workload away from larger VAX computers by processing files, creating graphics, or working with spreadsheets, "off-line", while the host is free to meet the demands of other users. Additionally, resources such as high speed printers, plotters, and centralized storage, of the VAX can also be utilized and shared by many Macintosh users. A basic understanding of file transfer techniques will help to open up the communications between your Macintosh and VAX systems.

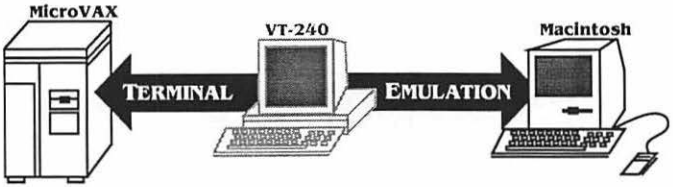
TERMINAL COMMUNICATIONS

Most users of large computer systems interact with their host computer with a specialized terminal designed to communicate with that specific computer system. These terminals take input from the user, as it is typed on the terminals keyboard, and then transfer it to the host computer. Output from the computer is displayed back to the user on the terminals video display.

Years ago, VAX users had to have dedicated terminals to connect to their VAX systems. These terminals, namely the DEC VT100 and VT200 series, were specifically designed to work directly with VAX systems. Macintosh owners who need to access VAX computers no longer need to have a dedicated DEC terminal.

TERMINAL EMULATION

A Macintosh with appropriate software can take the place of (or emulate) a DEC VT series terminal. A *terminal emulator* is a program on a personal computer that sends and receives information in the same way as the "real" terminal. It simply pretends to be the terminal that the host system is expecting. The host computer doesn't know the difference.



The major function of a terminal emulator is to make the personal computers keyboard and display function as closely as possible to the terminal it is emulating.

Terminal emulation software running on personal computers, such as the Macintosh, has many advantages over straight terminal to host connections. One of the biggest advantages of emulation is that the Macintosh resources can be used in addition to the specific terminal features.

Common emulator functions such as, storable connection settings, review buffers, and file transfer capabilities, take advantage of special Macintosh features.

In fact, few software emulators limit themselves to only the capabilities of the terminal they are emulating. For example, while White Pine Software's Mac220, Mac240, and Mac241 products emulate the text and graphics modes of the DEC VT220, VT240, and VT241 terminals, they also include capabilities unavailable on the terminal itself. Automatic dialing and connection features allow users to customize their communication environments, and automatic logging with review buffers lets them capture information displayed during the terminal session. Perhaps the major advantage of terminal emulation is the fact that files can easily be transferred between the Macintosh and the VAX. Also, host systems usually have communications hardware, such as dial-in phone lines and modems, already in place. This takes advantage of the unique processing capabilities of the Macintosh while freeing valuable CPU time on the VAX.

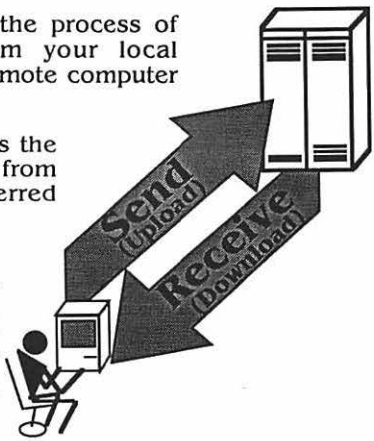
SENDING/RECEIVING

Two important aspects associated with file transfers are the concepts of *sending* and *receiving* files. Sending and receiving are also often referred to as *uploading* and *downloading*.

Sending, or uploading, is the process of transferring information from your local computer (a Macintosh) to a remote computer (a VAX for example).

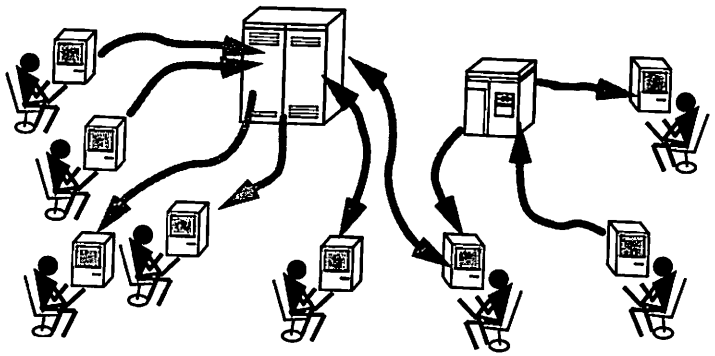
Receiving, or downloading, is the reverse process... information from the remote system is transferred down to the local system.

Since uploading and downloading are rather ambiguous terms that can be misunderstood, it's probably best to think in terms of sending and receiving information between local and remote systems.



HOST BASED FILE STORAGE

One important aspect of file transfers is that the extensive resources of host systems can be used to store and distribute information among many personal computer users. Host based file transfer applications are programs which handle the transfer, storage and retrieval of files sent between computer systems. These programs act as centralized filing cabinets for storing and maintaining the files of many users. Programs of this type work by performing file transfers with terminal emulators, and by providing the user with file maintenance capabilities. White Pine's VMacS product is a DEC VAX based file transfer and storage application. It runs under the VMS operating system on the VAX, and allows Macintosh users to store and share files in several formats in standard VAX/VMS directories. With this program, Macintosh users can store their files with various levels of protection or access by other users.



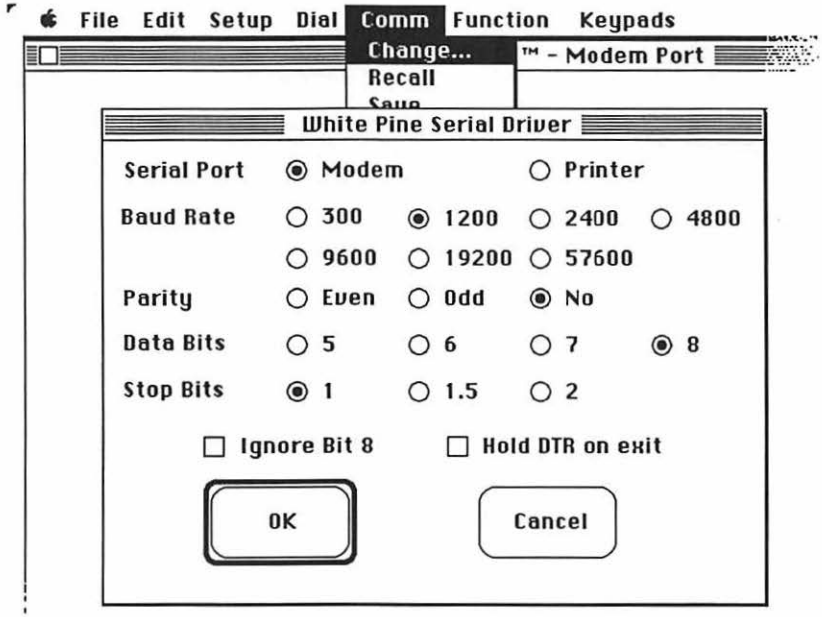
One of the benefits of VAX based transfer applications is that stored Macintosh files can be periodically archived under routine VMS backup procedures. By storing files in either Macintosh or host VAX compatible formats, file transfer applications enable users of both environments to access much of the same information.

Host based file transfer applications should not be confused with network based file servers, which operate within Macintosh networks to store and maintain user files.

EXAMPLES

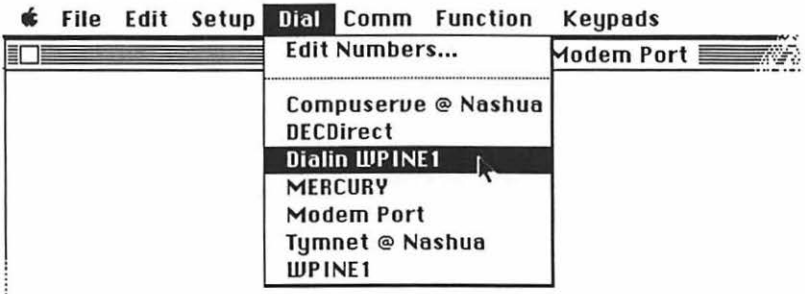
TERMINAL EMULATION EXAMPLE

Through terminal emulation, we can access our host VAX system from the Macintosh as if we were using an actual VT series terminal. For this part of the example, we'll perform the dial in connection from a home Macintosh to the VAX at work, just as in the example under "File Transfer Concept". Host connections using most terminal emulators are fairly simple once the communications parameters have been set. These parameters govern how the Macintosh will handle its side of the communications with the VAX. The dialog below shows how Mac241 was set up for our example VAX connection.

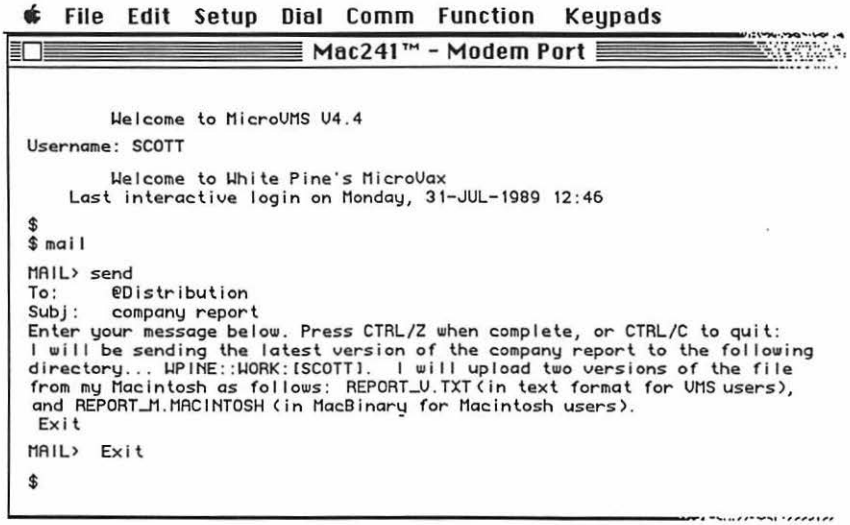


For now, the values for these parameters are not extremely important. The "Communications Parameters" section found later in this booklet explains most of these in detail.

After the communications settings have been set up to match those of the VAX, we simply use the dial menu to initiate the connection.



Once the Macintosh to VAX connection has been made, we can use the VAX as if we were using an actual VT series terminal. To illustrate this, let's distribute a mail message on the VAX via Macintosh terminal emulation.

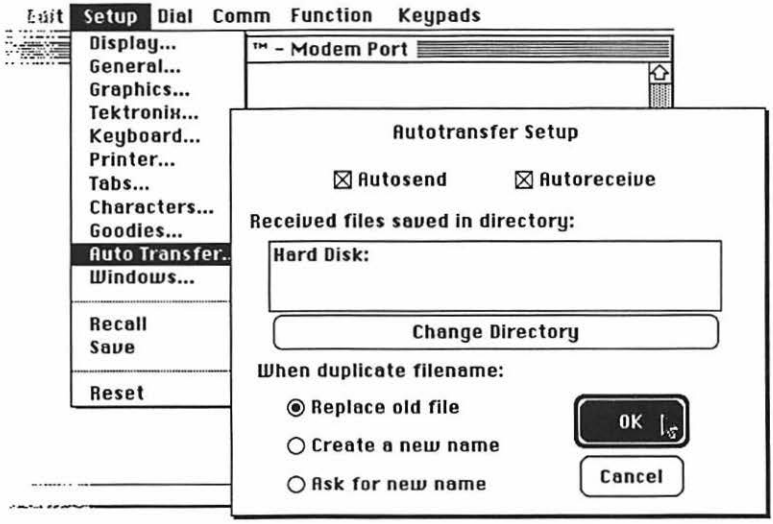


AUTO-TRANSFER EXAMPLE

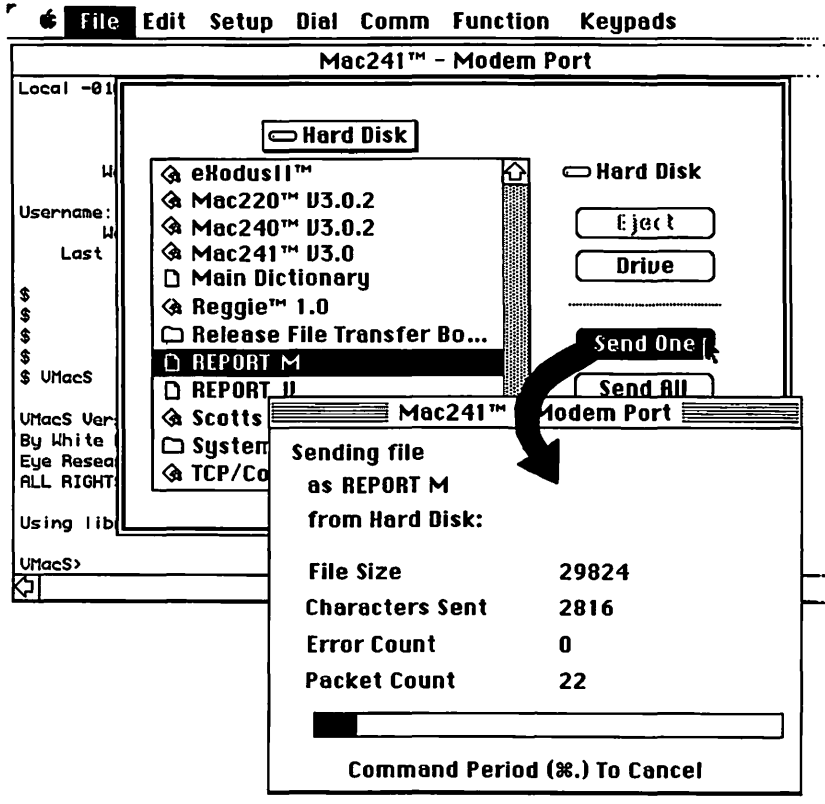
The next set of the examples illustrates a few ways to perform the previously mentioned transfers. Since we already have a connection to the VAX through our terminal emulator, all that's necessary is to set up the way in which the transfer will be handled by each system. The VMacS application will be used here to handle the VAX end of the transfers while Mac241 handles the Macintosh end.

One of the useful features available to users of White Pine's terminal emulators and VMacS is automatic file transfers. With auto-transfer set up between the two applications, file transfers can be performed quite easily. The only thing that usually needs to be done is to choose the file to transfer. The receiving program will automatically adjust itself to store the file in the appropriate format.

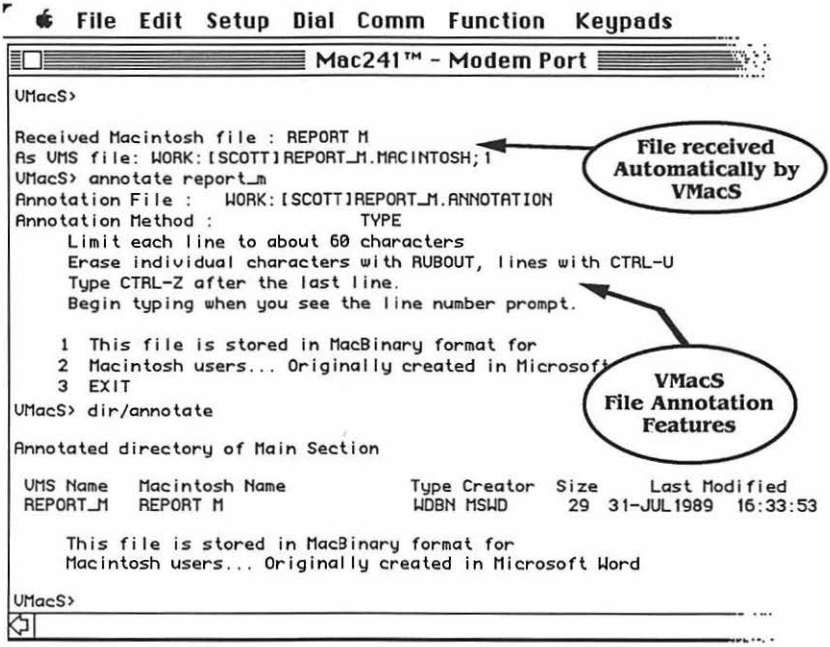
In this example we'll transfer the MacBinary version of our company report automatically to the VAX. The Mac VT series emulators have built in provisions for setting auto transfers, as shown below.



The next step is very simple, all we have to do is be in VMacS to receive the file while initiating the transfer from Mac241. This is accomplished by choosing the "File Transfer..." option under Mac241s "File" menu, and selecting the appropriate file from the dialog. From there, the transfer is handled automatically.



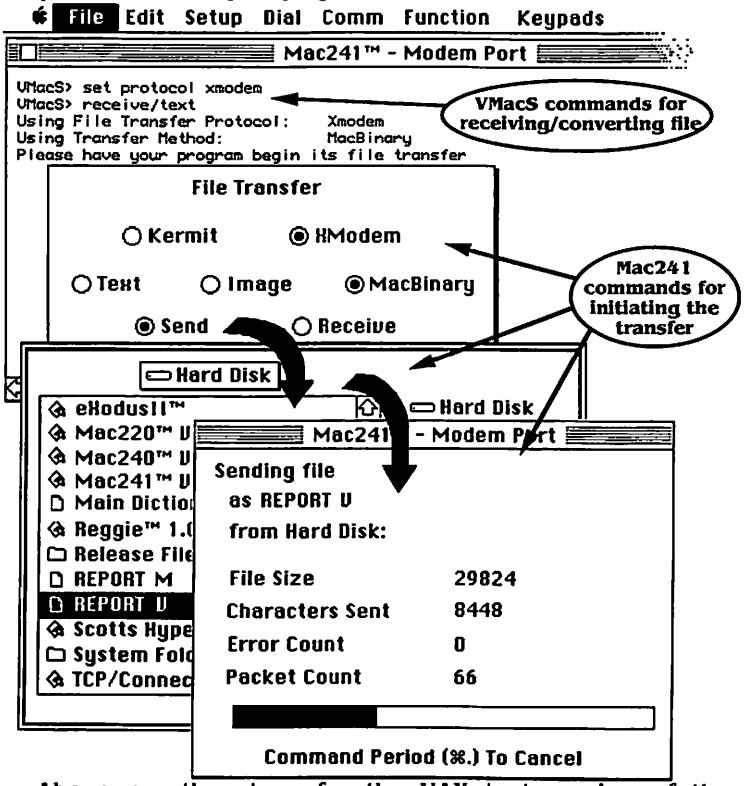
As seen below, VMacS automatically receives the file and stores it in the correct format. VMacS automatically accepts the file as MacBinary and adds the .MACINTOSH extension to the filename. Special characters and spaces within the Macintosh filename are also automatically replaced with the "_" underscore character.



The screen shot above also illustrates the file annotation feature of VMacS. This enables the user to enter descriptive information about the file. The file description can then be viewed by other users with the VMacS "Directory/Annotation" command. This, along with the various "Directory" qualifiers is very usefull in keeping track of files transferred and stored on the systems.

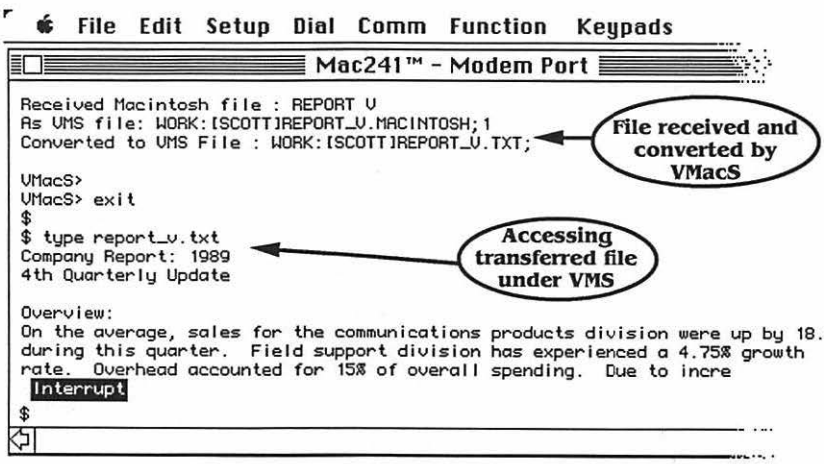
SPECIFYING TRANSFER SETTINGS

While auto-transfers are simple to perform, sometimes there may be a need to specify exactly how the transfer should take place. The rest of this booklet helps to explain how the communications for both systems are set up to transfer files, but for now we'll just illustrate some of the steps involved in specifying file transfer.



Above are the steps for the VAX text version of the Macintosh report. The VMacS commands "set protocol XMODEM" and "receive/text" were used to specify how the MacBinary file was transferred and converted to a VAX compatible text format. The settings for the VAX were set under VMacS first, and then Mac241 initiated the transfer.

Once the file has been transferred, it can be accessed by VAX users as a text file in the usual manner under VMS. For example, it can be edited in the EDIT application on the VAX, printed, or "Typed" to the display as shown here.

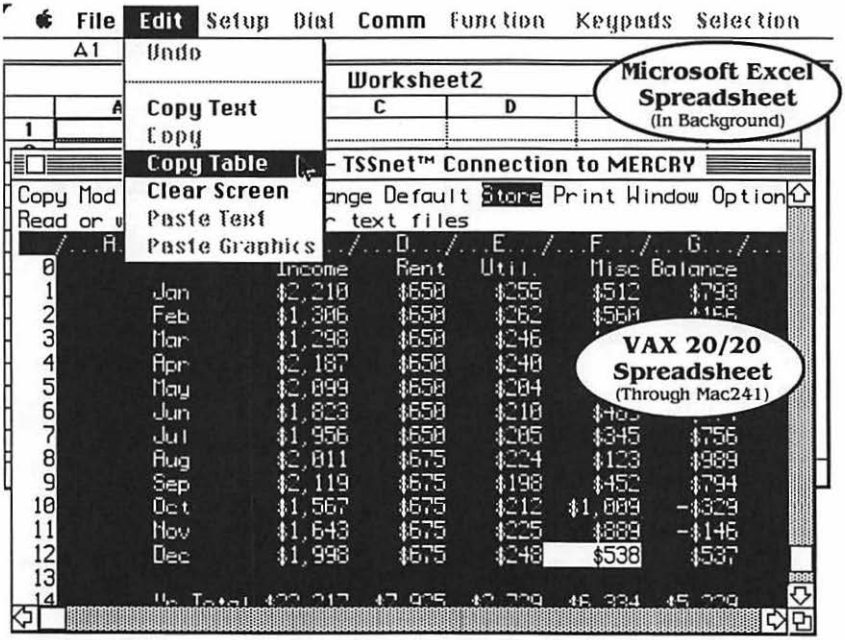


Text files transferred between systems often lose much of their formatting information. For the most part, line feeds, tabbing, and paragraph spacing gets "messed up" because the systems handle text formatting in different ways. While it is beyond the scope of this booklet to fully explore all of the features of VMacS, it should be noted that the "Table" and "Skip" qualifiers used in VMacS transfers allow files to be transferred and converted in a way that retains some of the original tabbing and paragraph separation.

CLIPBOARD TRANSFER EXAMPLE

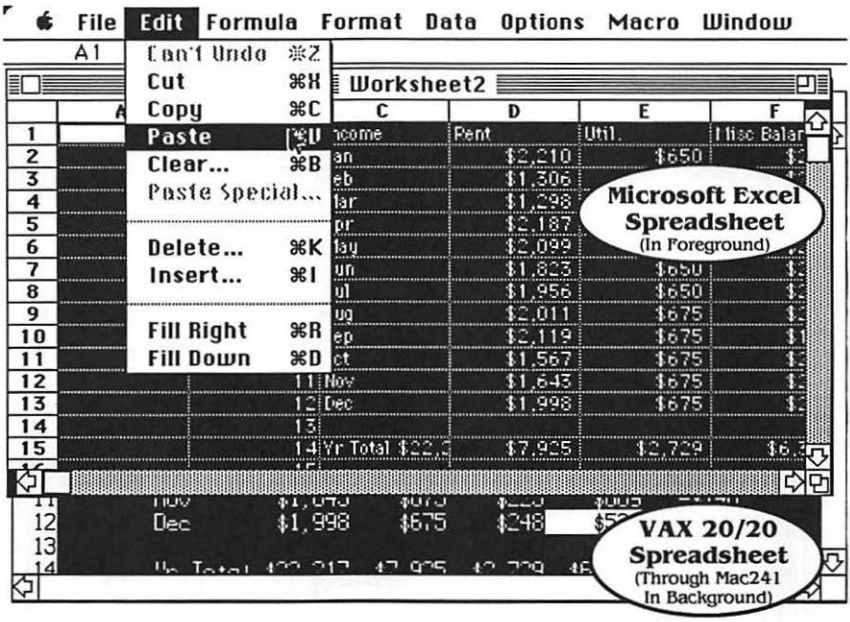
For another example of transferring between Mac & VAX systems, let's transfer some information from a VAX 20/20 spreadsheet for use within Excel on the Macintosh. This can be very easily done under Multifinder by logging on to the VAX with a terminal emulator to run 20/20, and simultaneously running Microsoft Excel.

Once both applications are up and running, cells from 20/20 can easily be selected and copied under the emulator, and pasted into Excel as illustrated on the next few pages.



In the screen shot above, we have both Mac241 and Excel running under MultiFinder. For this example, a budget spreadsheet was created on the VAX under the 20/20 spreadsheet application. Through terminal emulation with Mac241, it's a simple task to select lines of text from 20/20 and use the emulators copy functions to store the information in the Macintosh "clipboard".

The 20/20 spreadsheet information can now be transferred directly to Microsoft Excel on the Macintosh. This is done by clicking on the Excel application window to bring it to the front, and then pasting the clipboard into the spreadsheet, as shown next.

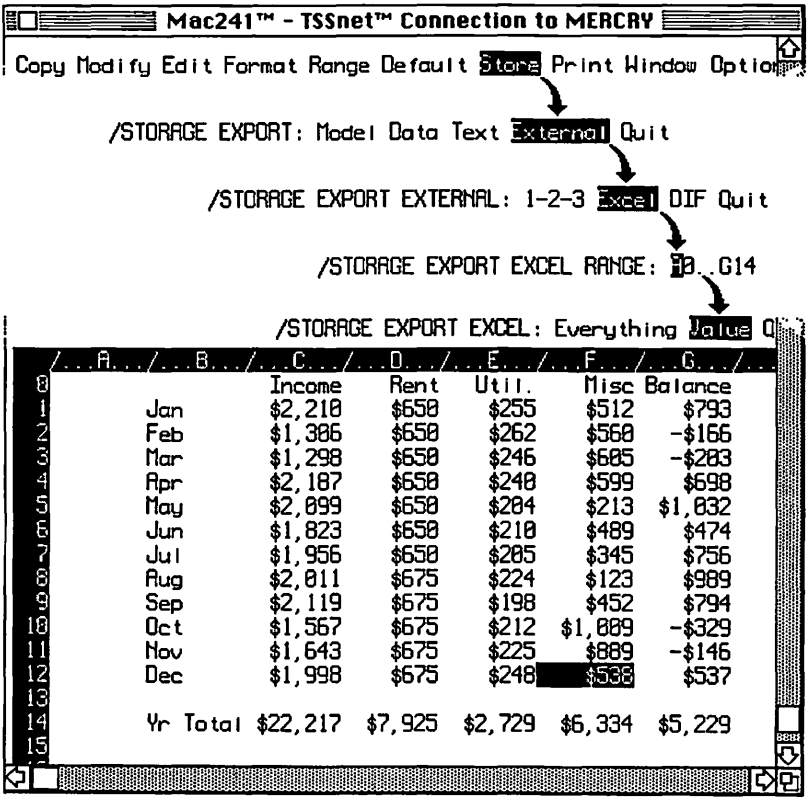


Even though copying sections from one spreadsheet to another in this manner is fairly straightforward, it would prove to be very time consuming for large spreadsheets. Another problem with this method is that it doesn't readily transfer formulas and calculations from one application to the other. If you only have a little information to transfer, this method is probably the easiest to perform. In fact, this technique can be used to transfer information between many Macintosh and VAX applications.

20/20 EXPORT TO EXCEL EXAMPLE

An even better way to transfer the information is to convert it to a format that Excel can use directly. This can be accomplished by saving the file in "Excel" format from 20/20, and then performing a file transfer to the Macintosh. After the file has been transferred, it can be opened and used in Excel just as if it were created on the Macintosh.

The first step is to save the 20/20 file on the VAX in an Excel compatible format. Fortunately, 20/20 has provisions for this type of save in its "Export" functions. This is illustrated below in the following 20/20 menu selections.



	A	B	C	D	E	F	G
0			Income	Rent	Util.	Misc	Balance
1		Jan	$2,210	$650	$255	$512	$793
2		Feb	$1,386	$650	$262	$560	-$166
3		Mar	$1,298	$650	$246	$685	-$283
4		Apr	$2,187	$650	$240	$599	$698
5		May	$2,099	$650	$284	$213	$1,032
6		Jun	$1,823	$650	$210	$489	$474
7		Jul	$1,956	$650	$285	$345	$756
8		Aug	$2,011	$675	$224	$123	$989
9		Sep	$2,119	$675	$198	$452	$794
10		Oct	$1,567	$675	$212	$1,089	-$329
11		Nov	$1,643	$675	$225	$889	-$146
12		Dec	$1,998	$675	$248	$538	$537
13							
14		Yr Total	$22,217	$7,925	$2,729	$6,334	$5,229
15							

All that needs to be done now is to transfer the file from the VAX to the Macintosh. This is similar to the previous example, where we transferred files from the Macintosh to the VAX. The process will be the same, only in reverse.

If we exit to the VMS level from 20/20, and run VMacS, we can set up the VAX side of the communications, and initiate the transfer.

```
Mac241™ - TSSnet™ Connection to MERCRY
BUDGET.W20;3          BUDGET_MAC.XLS;2     LOGIN.COM;8
Total of 8 files.
$
$ vmacs

VMacS Version 2.0 Copyright (c) 1986,1987
By White Pine Software and
Eye Research Institute of Retina Foundation
ALL RIGHTS RESERVED

Using library :                    USER$1:[SCOTTL]

VMacS> set protocol XMODEM
VMacS> send/foreign budget_mac.xls
```

VMacS is now ready to transfer the file to the Macintosh. At this point, don't worry about the protocols and formats used to set up the transfer. They're all explained in more detail later in other parts of the booklet. The following Dialog boxes illustrate the Macintosh end of the transfer through Mac241.

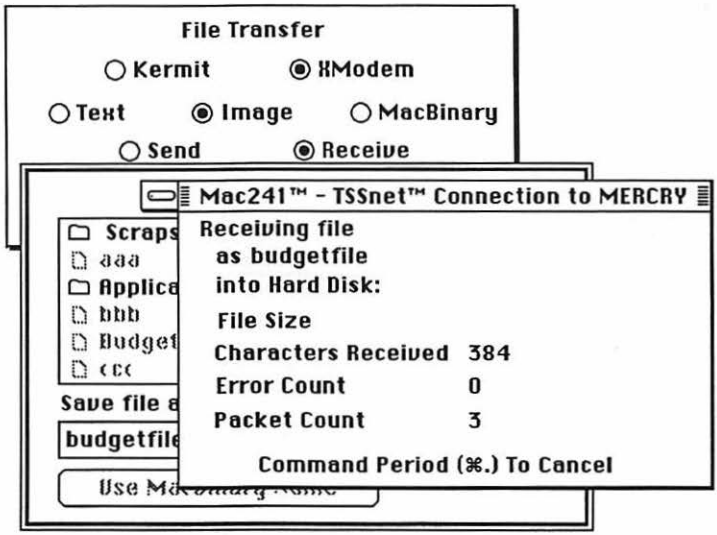


TABLE TRANSFER EXAMPLE

While the previous example has helped to illustrate the compatibility between the VAX 20/20 and Macintosh Excel applications, many VAX applications don't have provisions for saving files in Macintosh compatible formats. For example, although the information from a printable VAX/VMS report file is spaced in column format, the individual data items may not be properly aligned when transferred to a Macintosh database or spreadsheet. The following example illustrates a way of converting and transferring files with column formatting, through a special VMacS feature known as a *table transfer.*

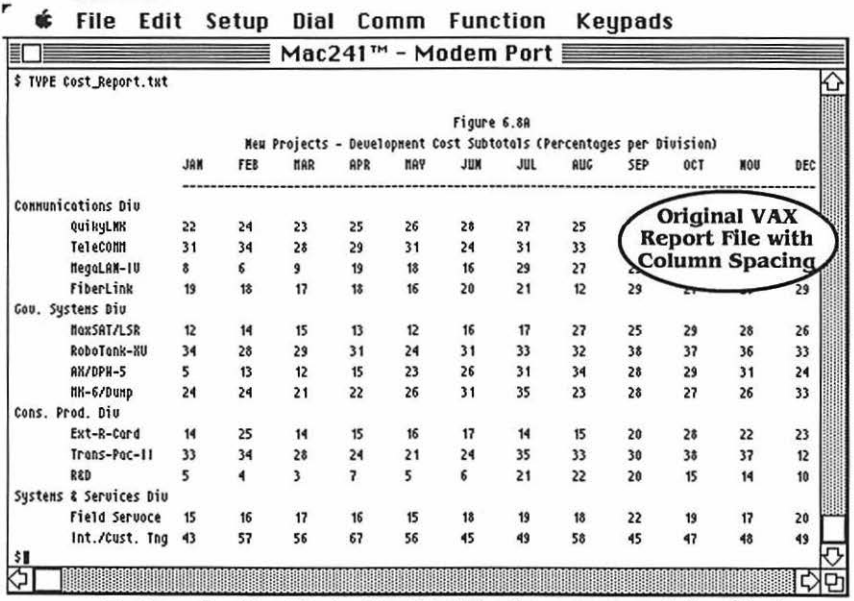


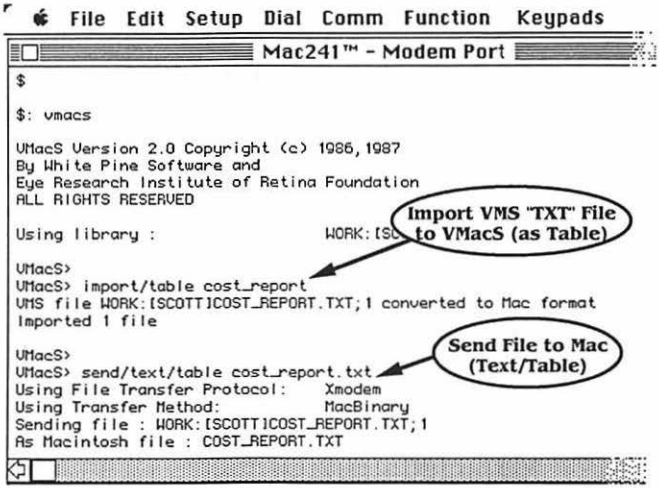
Figure 6.8A
New Projects - Development Cost Subtotals (Percentages per Division)

	JAN	FEB	MAR	APR	MAY	JUN	JUL	AUG	SEP	OCT	NOV	DEC
Communications Div												
QuikyLNK	22	24	23	25	26	28	27	25				
TeleCOMM	31	34	28	29	31	24	31	33				
MegaLAN-IU	8	6	9	19	18	16	29	27				
FiberLink	19	18	17	18	16	20	21	12	29			29
Gov. Systems Div												
MaxSAT/LSR	12	14	15	13	12	16	17	27	25	29	28	26
RoboTank-XU	34	28	29	31	24	31	33	32	38	37	36	33
AX/DPH-5	5	13	12	15	23	26	31	34	28	29	31	24
MM-6/Dump	24	24	21	22	26	31	35	23	28	27	26	33
Cons. Prod. Div												
Ext-R-Cord	14	25	14	15	16	17	14	15	20	28	22	23
Trans-Pac-II	33	34	28	24	21	24	35	33	30	38	37	12
R&D	5	4	3	7	5	6	21	22	20	15	14	10
Systems & Services Div												
Field Service	15	16	17	16	15	18	19	18	22	19	17	20
Int./Cust. Trg	43	57	56	67	56	45	49	58	45	47	48	49

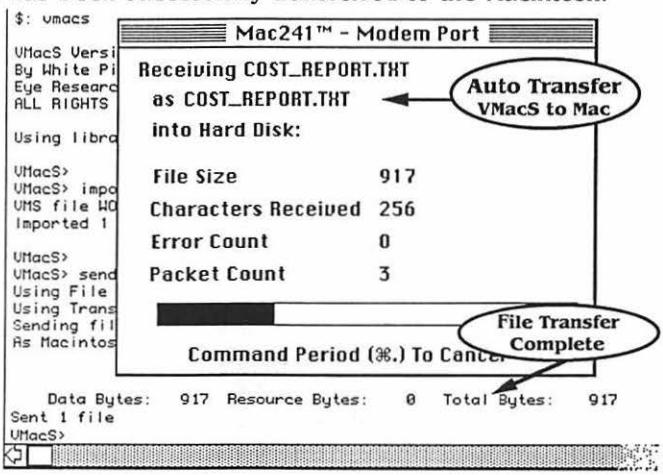
$

To transfer the file to the Macintosh, we convert the file to a Macintosh compatible format using the "Import" command under VMacS. The "/Table" qualifier causes VMacS to change the column formatting to Macintosh compatible table format. Multiple spaces or spaces followed by numbers are replaced by tab characters. When sent with other transfer methods that do not perform this conversion, the data may not properly align in the cells of a spreadsheet. The file transfer process is illustrated on the following page.

The screen shot below illustrates the steps involved to import and send the text file as a table.



Since we are using the auto-transfer feature of Mac241, the file is automatically received and stored at the Macintosh end. VMacS indicates that the process is complete when the file has been successfully transferred to the Macintosh.



The transferred file can now be accessed from applications on the Macintosh. Most Macintosh spreadsheet and database applications will accept table data in this format. Here, we've opened the file under Microsoft Excel. As you can see, the data in each column of the original VAX application file has been placed in individual cells of the Macintosh Excel file. Other transfer methods would have treated the data as long strings of text.

File Edit Formula Format Data Options Macro Window

A1

COST_REPORT.TXT

	A	B	C	D	E	F	G
1						Figure	6.8A
2					New Projects - Development Cost Subtot		
3				JAN	FEB	MAR	
4							
5	Communications Div						
6		QuikyLNK	22	24	23		
7		TeleCOMM	31	34	28	29	3
8		MegaLAN-IV	8	6	9	19	1
9		FiberLink	19	18	17	18	1
10	Gov. Systems Div						
11		MaxSAT/LSR	12	14	15	13	1
12		RoboTank-XV	34	28	29	31	2
13			5			13	

Transferred Report File In Excel

With a little bit of editing, the file can be enhanced to take advantage of the Macintoshes graphic capabilities.

File Edit Formula Format Data Options Macro Window

4R x 3C

New COST REPORT

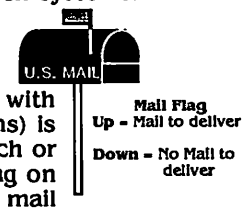
	A	B	C	D	E	F	G	H	I
1								Figure 6.8A	
2			New Projects - Development Cost Subtotals						
3			JAN	FEB	MAR	APR	MAY	JUN	JUL
4	**Communications Div**								
5		QuikyLNK	22	24	23	25	26		
6		TeleCOMM	31	34	28	29	31	2	
7		MegaLAN-IV	8	6	9	19	18	16	29
8		FiberLink	19	18	17	18	16	20	21
9	**Gov. Systems Div**								
10		MaxSAT/LSR	12	14	15	13	12	16	17
11		RoboTank-XV	34	28	29	31	24	31	33
12		AX/DPW-5	5	13	12	15	23	26	31
13		MK-6/Dump	24	24	21	22	26	31	35
14	Comm. Prod. Div								

Enhanced Report

TERMINOLOGY & CONCEPTS

Now that a few examples of the uses of file transfers have been given, some basic computer communication concepts should be explained. The next several pages will help to clarify some of the terms and concepts that you may encounter when talking about transferring information between systems.

BITS



The most basic element associated with computers (and computer communications) is the *bit*. A bit can be thought of as a switch or flag that tells us something. Just as a flag on a mailbox tells the postman that there is mail to be delivered, a bit is used to represent useful information in the computer.

Bits exist in two states... "Off" or "On" ("False" or "True", "0" or "1"). Computers turn on and off bits to represent and manipulate information.

Mouse Button Bit
0 = Button Up
1 = Button Down

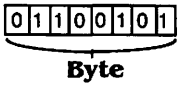
One simple example of the usage of a bit is with the Macintosh mouse button. The button has two states, up or down (or off and on). When the button is held down, a corresponding bit is "set" to the on position. Software applications running on the Macintosh can check the status of this bit, and act accordingly.

Bits which are "off" are typically represented in the computer as a "0", while bits which are "on" are represented by a "1". A single bit, by itself, has limited uses because it can only represent two states (0 or 1).

The power of computing really comes from the use of sequences of bits rather than single bits. Sequences of bits can be used to count numbers, store text, or manipulate any information on the computer. These sequences of bits are usually grouped together in units called *bytes*, *packets*, and *files*. These terms are commonly used when talking about file transfer.

BYTES

A *byte* usually refers to a sequence of eight bits. Bytes are the second smallest unit of information typically manipulated by computers.



While bits can represent two states (0 or 1), a byte can represent information which is much more meaningful such as a number, a character, a color, or virtually anything else used by the computer.

Bytes are also used as a unit of measure for computer memory. Usually people talk about having a computer with 512k, or 1Meg of memory, and double sided floppy disks that hold 800k, or hard disks with 40Meg. What they are talking about is the number of bytes involved. The terms "k" and "Meg" are short for kilobyte and megabyte, where 1 kilobyte ≈ 1000 bytes, and 1 megabyte ≈ 1000 kilobytes or roughly 1,000,000 bytes.

The term roughly and the ≈ symbol (which means approximately equal to) are used here because in computer terminology, a kilobyte actually represents 1024 bytes rather than 1000 bytes, and a megabyte means 1024 kilobytes rather than 1000 kilobytes. This is due to the way in which computers internally work with numbers in binary (which is discussed in the "Number Bases" appendix at the back of this booklet). In most cases it's acceptable to think in terms of the approximate values.

So when someone say's he is going to upload a 100k file from his Mac to the VAX, he is really saying that he is going to transfer about 100,000 bytes of information between the two computers.

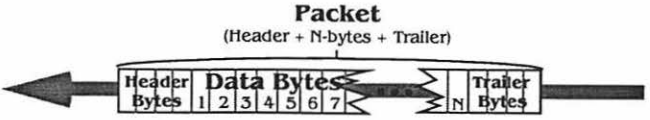
Most file transfers between computer systems involve the sending or receiving of characters. Characters are the basic elements of text files, and represent letters, numbers and formatting information. A character of text can be represented by 7 or 8 bits (or 1 byte). In the case of 7 bit characters, the extra bit can be used for simple error checking during the transfers.

PACKETS

In file transfers, bytes of information are grouped together and transferred in larger units that are easier to work with.

The term *Packet* is commonly used to describe these larger units. Packets consist of sequences of bytes which are transferred as a single unit. Special bytes are added to the start and end of each sequence. These extra bytes are referred to as *headers* and *trailers*.

The header and trailer information often describes characteristics of the data being transferred, and can contain information that is needed during the transfer, such as source and destination network addresses, routing information, the type of data, sequence numbers, and special error checking information.



A good analogy for packets can be found in a fictitious "widget" company. If we had to send widgets (the data bytes) to our retailers, we wouldn't just throw them into the back of a truck and send them on their way. We'd put them in cartons, cases, or some other larger packages that were easier to move and keep track of. We'd also attach shipping labels and other markings containing information such as type and lot number of the widgets, and retailer destination number. Time and effort could be saved by not labeling the cases, but there is a risk that some of the shipment could be lost or misplaced. The labeling helps us keep track of the items sent and received. The header/trailer information of packeting works in a similar way.

FILES

All of the programs, documents, and folders on the Macintosh, as well on the VAX are commonly referred to as *files*. Files are just large sequences of bytes stored on disks. In a file transfer between two computers, the bytes of a file are sent from one computer and received by the other.

This can be done by just sending the bytes in sequence, or by using the blocking and packeting mechanisms of file transfer protocols. The "File Transfers" section will describe this in more detail.

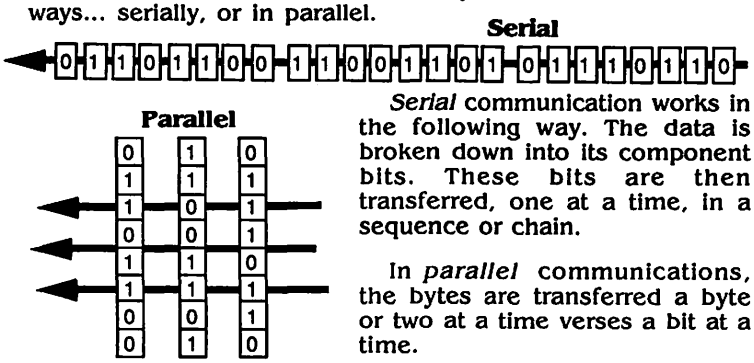
For the most part, when a file is transferred, it's information is transparent to the computer systems involved in the transfer. This is analogous to the mail system, where the content of a letter or package is not important. The post office will deliver the package regardless of the information it contains, as long as it meets the size or weight restrictions, has proper postage and addressing information. Text files are an exception to this, in that formatting information, such as carriage returns and line feeds, can be added or deleted to make the file compatible with format of the receiving system.

COMMUNICATIONS & CONNECTIONS

There are several different ways to connect computers together. This section briefly describes ways in which personal computers, such as the Macintosh, can connect to, and communicate with larger host computers like the VAX.

SERIAL/PARALLEL

At the lower levels of computer communications, the bytes of a file are transferred between systems in either of two ways... serially, or in parallel.



Serial communication works in the following way. The data is broken down into its component bits. These bits are then transferred, one at a time, in a sequence or chain.

In *parallel* communications, the bytes are transferred a byte or two at a time verses a bit at a time.

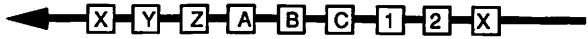
In general, serial paths are used for telecommunications and networking, while parallel connections are primarily used for internal bus communications and connections to hardware devices such as disk drives and printers. The Macintosh modem and printer ports send and receive data serially, while the SCSI port, which usually connects external disk drives to the Mac, sends and receives in parallel.

SYNCHRONOUS & ASYNCHRONOUS COMMUNICATIONS

Computers have two different ways of transmitting a series of characters.

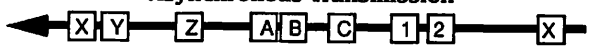
In *synchronous* communications, characters are sent between the systems at precise intervals. Synchronous communications require that the internal clocks of the two computers (or modems) remain "in step" with each other. The systems keep in "sync" with each other by sending a special signals during the transmission.

Synchronous Transmission



In *asynchronous* communications, the time interval between the characters is not fixed. A person typing at a terminal is a good example of asynchronous communications. In terminal communications, the user doesn't enter characters at a uniform speed. So, the time between two transmitted characters will not be the same each time. Asynchronous transmissions often require the use of special markers that signal the start and end of each character (see "Start & Stop Bits" later in this booklet for more information).

Asynchronous Transmission



COMMUNICATIONS PARAMETERS

The software that enables your Macintosh to talk to VAX systems needs to be told how to set up communications between the two systems. Terminal emulators, and other applications that transfer information between systems often have "Settings" or "Communications" menus that allow users to specify how communications are to be handled. This section will cover the common communication parameters used in programs of this type.

DATA BITS

The term *data bits* (also referred to as *word length*) refers to the number of bits needed to send for each character. The number of data bits can range from five to eight, with most systems using seven or eight.

As an example, the letter A is represented on the Macintosh by the number 65, or by the binary digits 01000001. This character can be represented by 7 or 8 bits. Other characters, such as special control codes, often require all 8 bits, so the number of data bits must be set to 8 to properly transmit them. Most Macintosh to VAX communications use 8 bits for this parameter.

START & STOP BITS

Asynchronous communications need to send special (non-data) bits along with each character to mark the start and end of the character. In order to have the receiving system recognize the start of a character, a start bit is inserted before each character. Some systems may also require a number of stop bits to be placed at the end of each character for proper transmission. One stop bit is usually specified for Mac to VAX communications.

PARITY

Computers can check for communication errors by setting the highest bit of each transmitted character a certain way.

There are several different types of parity specifications. In *even parity* the highest bit is set or cleared to make the total number of "on" bits in the character an even number. *Odd parity* is just the opposite, the parity bit is used to make the total number of "on" bits an odd number. In mark parity and space parity, the highest bit is always turned "on" or "off" respectively. *No parity* simply means that no parity bits are generated.

Parity works in the following way... if two computers are both set up with odd parity, and one computer is going to send the 7 bit character A, its bits are set as 01000001 (65 decimal or 41 hex). The sending computer counts the total number of "on" bits in the character (2 in this case) and sets the parity bit so that the total "on" bits is an odd number. The transmitted byte now becomes 11000001. The computer at the other end of the transmission receives the byte, and seeing a set parity bit, knows that the total number of on bits must be an odd number. Since it is, the computer knows that the byte is good.

Some file transfer protocols, such as XMODEM require an 8 bit channel and cannot use the eighth bit for parity. "No parity" is commonly used during Mac to VAX communications.

BAUD RATE

The speed at which computers communicate is often referred to by the terms *baud rate* or *data rate*. Most people think of baud rate as the number of bits per second that can be transmitted between the systems. This, however, is not always a correct definition of the term.

When digital information is transmitted over analog lines, each second is divided into smaller time intervals. Normally, devices transfer 1 bit of data per interval by sending one type of signal to represent zeros, and another type of signal to represent ones. More than one bit can be transmitted during each time interval if more than two different signals are used. A device that could send four voltage levels during each interval, for example, could represent the values 0, 1, 2, and 3 (or 00, 01, 10, and 11). This allows two bits to be transmitted in each interval. Signaling of this type are called a dibit transmissions, and cause the number of bits per second to be twice that of the specified baud rate. So, rather than being the number of bits per second, baud rate actually refers to the number of times that the signal can change per second. Data rate is actually the correct term for the number of bits per second. Unfortunately, baud rate is very often used interchangeably with data rate because they're usually equivalent.

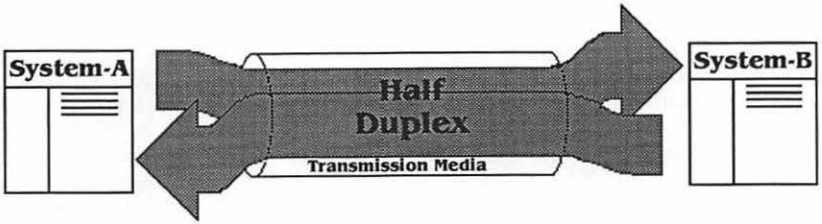
Typical data rates are : 300, 1200, and 2400 for telecommunications, and 4800, 9600, 19200, and 57600 for networks and direct lines. The important thing to remember is that both systems must communicate at the same speed. If one system is sending and receiving at 1200 baud, it will be able to communicate with other systems working at the same baud rate. It will get garbage instead of data if it tries to communicate with a device transmitting at 9600 baud.

Another concept that is important during file transfers, is *characters per second*. As a general rule of thumb, the number of characters transmitted each second can be calculated by dividing the data rate by 10.

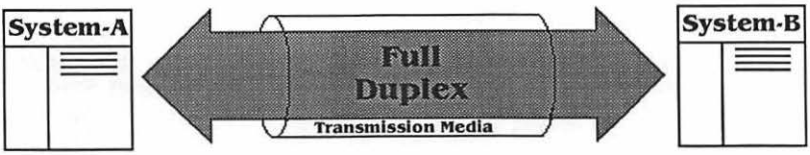
This works because most characters are 8 bits with a start and stop bit at either end. For example, a system transmitting at a data rate of 9600 bits per second can send approximately 960 characters each second.

DUPLEX

Computing devices can send signals to each other over the physical transmission medium in two ways. One way is to have one system send its information over the line, while the other waits for the first to stop before it transmits. This is known as *half duplex* transmissions. Information can move between the two systems in both directions, but only one way at a time.



Full duplex communications enable both systems to transmit their information at the same time. Information can move in both directions simultaneously. Full duplex transmissions avoid the delays involved with waiting for the transmission direction to change.



A concept that is sometimes confused with duplex is *local echo.* On DEC VT series terminals, for example, the terminal can be set to Local Echo, or No Local Echo, depending on whether or not the terminal should display all characters it sends locally, or wait to display characters as sent back by the host. This is useful in determining if characters are being altered somehow during transmissions.

PROTOCOLS

When computers communicate, they follow a specific set of rules that controls the exchange of information in an orderly manner. These rules are generally known as protocols. Protocols govern aspects of computer communication such as who is sending, who is receiving, when to start, when to stop, and how much information is transferred at a given time.

For an example of a protocol, think of two people in a conversation. Usually, when one person is speaking the other is listening. The listener, from time to time, acknowledges that he is receiving the information by nodding or saying "yes, go on..." or by using other subtle signals. The speaker also signals he is done by lowering his voice, or prompting the listener. Since eye contact and body movements are used so much, human communication is often far more complicated than computer protocols. People usually don't think of it, but they are following protocols when they talk with each other.

Often, protocols used in computer networks are "layered" to allow devices to handle the specific implementation of protocol software in their own way. The aspects of communication are broken down into functional layers. Each layer is responsible for handling the data at a certain level. For example, lower levels are responsible for handling data at the bit level and interfacing with the communications hardware, while upper levels handle the data in larger units, such as blocks, packets, records and files.

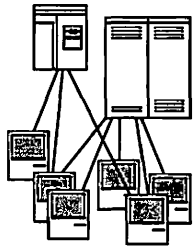
In general, the file transfer protocols used in asynchronous communications are fairly simple, and are not usually broken down into functional layers.

DIRECT CONNECTIONS

The most straightforward way to get two systems to communicate with each other is through *direct connections.*

This type of connection involves some sort of specialized cable that plugs directly into ports on both systems. Direct connections are simple if the computers are both the same model, or at least made by the same manufacturer,

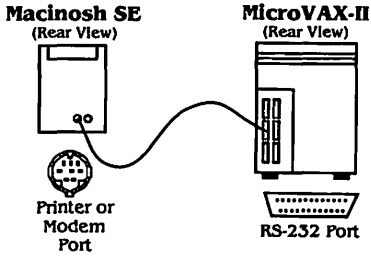
while connections of this type are often more difficult between computers from various vendors and with different architectures. This often requires special devices and adaptors to overcome the differences between the systems. Fortunately, the adaptors and cabling needed to connect most popular computers to each other is available through most data communications cable suppliers.



Direct connections often support the fastest physical transmission characteristics for transferring information between computers. This is due to the fact that there are usually fewer conversions and intermediate devices involved in the transfer.

While direct connections offer speed advantages, they also have the disadvantage of having less flexibility. If a company has hundreds or thousands of computer users, it will also need hundreds or thousands of lines running through the ceilings and floors of the building if direct connections need to be used. This type of configuration is often difficult to manage and adapt to changing system requirements. Another disadvantage of direct connections is that the communications speed must be decreased as the length of the cable increases unless special hardware, such as short-haul modems, are used.

Direct Macintosh to VAX Connection

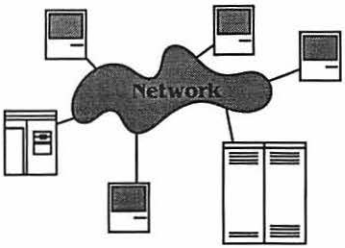


The Macintosh printer or modem ports, for example, can be connected directly to the VAX RS-232 ports with appropriate cabling. At the back of this booklet, the "Pinouts" appendix has more information on these ports.

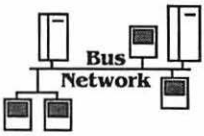
NETWORK CONNECTIONS

Another way to tie Macintoshes to VAX systems is through communication *networks*. Networking eliminates many of the flexibility and maintenance problems of direct cabling, and allows many systems to be interconnected with fewer lines.

Network Communications



Computer networks, (most commonly Local Area Networks or LANs) can have many configurations of both hardware and software. There are numerous type of cabling, or transmission media, used for networking purposes. Many LANs utilize the existing telephone cabling within a company, or use various forms of twisted pair wire, coaxial cable, or fiber optics to interconnect the systems. Depending on the network design, the connections can go from computer to computer, as in "ring" networks, or can exist as a single shared medium in "bus" networks. These are just two of the many possible network configurations.



The important thing to remember is that most networks are set up to give each computer (called Nodes) the ability to communicate with every other node on the network. Macintosh network communications products from Alisa Systems enable White Pine's Mac220, Mac240, and Mac241 emulators to have direct access to DECnet environments.

When computers communicate over a network, they follow specific network protocols which determine how information will be packaged, sent, and received. There are several networks and corresponding protocols that are currently popular for connecting Macintoshes to each other, and to VAX systems. The most popular include the LocalTalk and Ethernet network configurations using AppleTalk, DECnet, TCP/IP protocols. Since a detailed discussion of networking lies outside the scope of a booklet on file transfers, we've included networking vendors in the "For more Information" appendix at the back of this booklet. You can also check the computer section of your local bookstore for more information on networking.

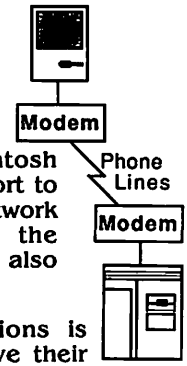
DIAL-UP COMMUNICATIONS - MODEMS

Telecommunications between computer systems involves using the existing telephone networks as the communication media. In our example under "File Transfer Concept", the user accessed a VAX from his Macintosh at home. This was accomplished by what's called a *dial-up* connection. The Macintosh used a *modem* and telephone connection to connect with the VAX.

Since telephone lines were designed to carry voice signals and not the digital information (0's and 1's) used by computers, special devices are needed to convert the digital computer signals from one computer into a form that can be transmitted over the phone lines. Modems are devices that do just that. A modem works by altering a phone signal so that it reflects the digital information of the computer. If two computers are equipped with modems, they can communicate over regular telephone lines as if they were directly connected together. The modems handle the task of converting the digital information to analog and back again.

The word *modem* comes from the terms modulate and demodulate. A modem modulates (changes) a carrier frequency in a way that reflects the digital data to send information, and demodulates the modified carrier frequency to receive and convert the data back to its original digital form.

When computers communicate over phone lines, the computers internal parallel data is converted to serial for phone line transmissions and back again at the receiving end.



Modem devices available to the Macintosh include modems that plug into the modem port to tie you're Mac directly to the phone lines. Network based modems, that enable any Mac on the network to connect to an outside line, are also available.

Another aspect of modem communications is compatibility. Modems that send and receive their information in the same way are said to be compatible. A standard that has emerged to be very popular is "Hayes". If you have a modem that is "Hayes" compatible, then you are generally ensured of compatibility, since most communication software supports "Hayes" compatible modems.

CONNECTION COMBINATIONS

The connection methods, just discussed, can also be used in combinations to connect Macintoshes to VAX systems. For example, a Mac can connect through a local Macintosh network to a network based modem, and then over the phone lines to a VAX host system with a modem. Network connections often use special bridge devices that enable transmissions to cross between different networks. Data transmitted over connections of this type can even involve transmissions over several types of media, such as coaxial cable, fiber optics, or telephone lines.

FILE TRANSFERS

Files can be transferred between two computers in two ways. They can be moved by using either stream transfers, or protocol transfers. The pages that follow describe some of the common ways in which these are handled.

STREAM TRANSFERS

One way of transferring a file from one computer to another is to send the characters in a stream. That is, one character is sent after another in sequence. To the receiving computer, the characters of the file appear just as if someone were typing them in. Stream transfers are fairly fast, since there is a minimum of extra information sent along with the characters. However, a drawback of stream transfers is that there are usually no, or few, provisions for error checking or recovery. Stream transfers also usually have no provisions for converting between the formats of each system. This can result in characters (or sequences) being misinterpreted or lost by the host.

PROTOCOL TRANSFERS

A more reliable way to send files between systems is by using file transfer protocols. Protocol transfers involve error checking and recovery procedures as well as other information (for example, file name, size, and type), which is useful for the transfer. Protocol transfers are, for the most part, slower than stream transfers because of this extra information and error checking, but are frequently used because they offer protection against loss of data during the transfer. With all protocols, if the sender and receiver both have the same protocol software, they can successfully transfer information, because they are both "speaking the same language". They are following the same set of rules that are defined by the transfer protocol.

XMODEM

Perhaps the most popular file transfer protocol is XMODEM. It was developed in 1977 by Ward Christensen for exchange between CP/M systems.

This protocol has evolved and been standardized to allow file transfers between a wide variety of computer systems. XMODEM is a simple protocol that is suited to transfers of binary files. It transfers data in fixed blocks of 128 characters at a time. XMODEM is also an eight bit protocol, meaning that it sends characters that consist of eight bits.

Most dial-up or direct connections should be fine with XMODEM, but problems may occur with systems that use special 8 bit control characters to manage communications. Terminal switching or network devices may have problems if they restrict data to 7 bits, or require 8 bit software flow control. XMODEM requires a *transparent* 8 bit line. If software flow control cannot be turned off, there will be problems during the transfer. If transfers consistently stop after a certain number of packets, this usually indicates that software flow control is active and should be disabled. XMODEM has several extensions that make it faster and more reliable. Newer protocols such as YMODEM and ZMODEM have also been based on the XMODEM protocol.

KERMIT

Kermit is another popular file transfer protocol, and was developed in 1981 at Columbia University. Kermit was designed to overcome some of the limitations of XMODEM to provide a more reliable environment for file transfers.

Kermit may be used over any communications medium between computers since it transfers characters that are seven bits in length. The use of seven bit characters allows files to be transferred without conflict or interception by communications equipment.

Look in the "For More Information" appendix for details on where to get more information on the XMODEM or Kermit file transfer protocols.

FILE FORMATS

Files can be transferred between Macintosh and VAX systems in several different file formats. The format that is used is dependent on what the files eventual end use will be. For example, files moved to the VAX for storage only, should be stored unchanged so that they can be downloaded back to the Macintosh in their original form. Conversely, if files moved to the VAX are going to be used by VAX users, then they should be converted from the Macintosh format to one that is usable on the VAX.

TEXT

Most computers have a file format that supports the storage of textual information, such as the words in this booklet. A text file is actually made up of bytes. These bytes represent the characters of the text. Several standards for representing characters on computer systems have been adopted. The most popular standard is known as ASCII (American Standard Code for Information Interchange). Both the Macintosh and VAX computers work with ASCII character sets, and have also use extensions to represent special characters.

In addition to letters, numbers, and other visible characters, text files also usually includes special bytes (control characters) within the text to specify formatting. The embedded characters are not visible in the text, but are used to mark carriage returns, page breaks, italics, and other format related information. Some formatting controls, such as end of line markers, are handled differently on various systems. When a text file is transferred from the Macintosh to another system, some of the formatting must be changed to be compatible with the new system. For example, White Pine's VT200 Series emulators add a line feed character to each carriage return. The receiving system can then either keep both the carriage return and line feed characters, or remove either one as needed to mark new lines.

IMAGE

Files that contain non-textual information, such as graphics, or data files, are stored and transferred between systems in a different format. Files in this format are called image files, or on some systems binary files. Whereas the content of text files can be modified to adapt to the formatting rules on each system, image files must be transferred with no changes at all.

MACBINARY

Files on the Macintosh are more complex than those on most other systems. Most Macintosh files are made up of three distinct parts: the file header, the resource fork, and the data fork.

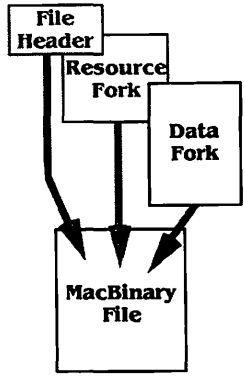
The purpose of the file header is to describe the structure of the file. It contains information such as the file name, file type, size, creator, and the date of the last modification.

The resource fork contains other information associated with the file. For data files, such as MacPaint graphics, or MacWrite files, the resource fork contains information need by the application. In Macintosh applications themselves, the resource fork is used to hold the program code.

The data fork usually contains the main body of information in a data file. For example, the text of a word processor document is probably stored in the data fork.

In most cases, the separate parts of a Macintosh file are transparent to the user. He just double clicks on a document or application to view a file or run a program. If a Macintosh file were to be transferred for use on a host system, most of the information in the file header and resource fork would be meaningless to the receiving system. Files that have been transferred for storage only need to retain that information to be useful on other Macintoshes.

The MacBinary format has been developed to easily move Macintosh files to other systems for storage. MacBinary format combines the three components of a Macintosh file into one single file. The MacBinary file can then be transferred, and stored, on another system that supports simpler file types. Later, the file can be transferred back to a Macintosh, where it can be separated into it's sub-components, and used as usual.



CONVERSIONS

Since most Macintosh files are made up of several distinct parts, a Macintosh file may undergo some modifications when transferred to another system. For example, a text file transfer involves only the data fork of a Macintosh file, with

certain modifications to control character formatting. An image file also transfers only the data fork, but the contents remain unchanged during the transfer. Since these formats do not contain all three components of Macintosh applications or data, MacBinary is used if the files need to be reused again on a Macintosh.

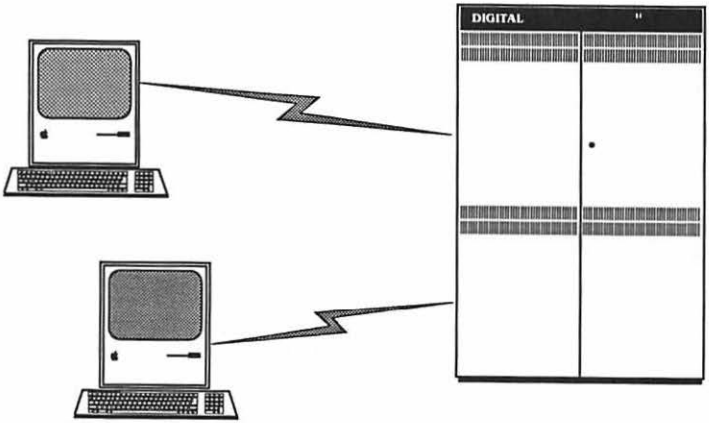
Most Macintosh terminal emulators, as well as host transfer programs, have provisions for transferring files in any of the formats just described.

When a file is transferred from one system to another, the information may not always be readily usable on the receiving system. If a MacWrite file is uploaded from the Mac to the VAX in MacBinary format, for example, it will be unusable as a text file in the VAX EDT editor. Since most terminal emulators and host based file transfer applications provide ways of transferring in each format, it is usually only necessary to choose the appropriate format for the file transfer. If a file is going to be stored on the VAX for access by other Mac users, then MacBinary should be used as the format for uploading and downloading. When a file is transferred from the Mac, to be used as a text file on the VAX, (or the other way around) then text format should be used. In a text transfer, the data fork of the Mac file is sent with changes to control characters, such as carriage returns and line feeds, by the receiving system. In an image transfer, the data fork of the Mac file is sent with no changes to its content.

Some forms of data, such as graphics, do not necessarily transfer correctly with any of these methods. There are many programs available that perform the format conversions required to make the transferred information useful on the receiving system. For example, "Reggie" from White Pine, is a Macintosh application that performs conversions between bitmapped graphics created with programs like MacPaint, or object oriented graphics such as those created with MacDraw, and their corresponding VAX file formats: "SIXEL", and "ReGIS".

SUMMARY

Ok, so now you're an expert on Mac to VAX file transfers... well maybe not an expert, but at least now you know a little more about what's going on behind some of those cryptic communication parameter and protocol settings. If problems do arise, you'll know what questions to ask and will be better able to describe your situation.



This booklet has been developed to give you an understanding of the most important aspects of Mac to VAX communications. However, a book this size can't realistically cover all situations and explain all of the concepts involved in great detail. In fact, we've only skimmed the surface on some topics. Please refer to the "For More Information" appendix, at the end of the booklet, for a listing of vendors and information pertaining to Mac to VAX communications. Visit the computer section of your local library or book store and pick up a text on computer communications if you're really looking for details. Also, talk with system managers and computer communications people that you know... they should be able to explain the specifics of your particular systems. If you still have difficulty with Mac to VAX file transfers, give us a call. Chances are, we'll be able to point you in the right direction and help you get the most from your Macintosh and VAX systems.

APPENDIX A: NUMBER BASES

People and computers process information in different ways. The human brain is capable of dealing with concepts that are vague or imprecise, while the computer can only handle information that is exactly defined. Because of this, the numbers that we use every day are also represented and used differently by computers.

You probably will not need to deal with binary or hexadecimal numbering when making most Mac to VAX file transfers, but since computers process all information (text, graphics, spreadsheets etc.) as numbers, an underlying understanding of how this works should be beneficial.

DECIMAL

We use the digits 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 to represent numbers, that is, our numbering system is based on ten symbols. So when we count, we use the sequence 0, 1, 2, ... 8, 9, and when we run out of symbols we mark that we have counted through the set by using two digits to make 10, and then start over with 11, 12, ... 19, 20 and so on. When we see a number like 38 we know that it represents 3 sets of tens and 8. This way of counting is easy for people, but difficult for computers whose internal memory and processing only works in the two states "off" and "on".

BINARY

Computers internally work with bits, which can only represent the two numbers 0 and 1. These two digits are used to make up all numbers in *binary*. Just as decimal numbers go through the set of 0..9 then start over with 10, 11... binary numbers go through the set 0, 1 then start over with 10, 11. This process continues to represent all binary numbers. To illustrate this, we'll count to 15 in binary (decimal in normal type with binary in bold)... 0=**0**, 1=**1**, 2=**10**, 3=**11**, 4=**100**, 5=**101**, 6=**110**, 7=**111**, 8=**1000**, 9=**1001**, 10=**1010**, 11=**1011**, 12=**1100**, 13=**1101**, 14=**1110**, 15=**1111**.

In actual use, large binary numbers, such as 1101011101001101, can be difficult to work with. Since memory chips and other computer components can only manipulate simple off-on elements, binary is the most efficient way for computers to store and manipulate information. If someone were to invent computer memory and processing elements that could easily process 10 different values instead of two, then computers could work directly in decimal.

HEXADECIMAL

Since working directly with binary numbers such as 1011101001110110 and 1110100011101011 is somewhat awkward for people, a simpler way to represent binary numbers has been developed. This simpler representation is called *hexadecimal.*

It works in the following way. If you take a binary number, such as 1100010011110110, and break it into sections of 4 bits, you get 1100, 0100, 1111, and 0110. This means that our example binary number can now be represented by the numbers 12, 4, 15, and 6, instead of 1100, 0100, 1111, and 0110. The only problem with using 0 through 15 is that we need to separate each number to avoid confusion. If we strung the numbers together, 124156, could be interpreted several different ways. Hexadecimal gives us a simple solution. Instead of the numbers 10, 11, 12, 13, 14, and 15, we use the letters A, B, C, D, E, and F to represent the 4 binary digits. So now, our example number is represented by the hexadecimal number C4F6.

It takes a little getting used to, but once you get the hang of it, working with the various numbering systems becomes easy. Remember people tend to think in decimal (base 10), while computers "think" in binary (base 2). Hexadecimal (base 16) makes binary easier for people to work with. Keep in mind that the content of the data never actually changes... just the way we look at it.

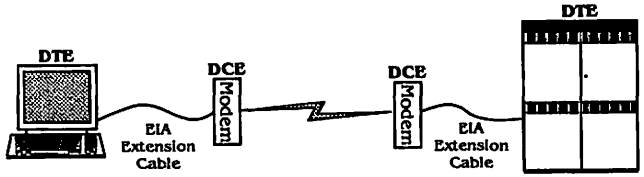
The chart below shows a list of numbers in each of the bases. Decimal is based on powers of 10, and uses the symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, and 9 to represent numbers. Binary is based on powers of 2, and uses only the symbols 0, and 1 to represent numbers. Hexadecimal is based on powers of 16 and uses the symbols 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, and F to represent numbers.

Decimal	Binary	Hex	Decimal	Binary	Hex
0	0	0	41	101001	29
1	1	1	42	101010	2A
2	10	2	43	101011	2B
3	11	3	44	101100	2C
4	100	4	45	101101	2D
5	101	5	46	101110	2E
6	110	6	47	101111	2F
7	111	7	48	110000	30
8	1000	8	49	110001	31
9	1001	9	50	110010	32
10	1010	A	.	.	.
11	1011	B	.	.	.
12	1100	C	.	.	.
13	1101	D	127	1111111	7F
14	1110	E	128	10000000	80
15	1111	F	.	.	.
16	10000	10	.	.	.
17	10001	11	.	.	.
18	10010	12	255	11111111	FF
19	10011	13	256	100000000	100
20	10100	14	.	.	.
21	10101	15	.	.	.
22	10110	16	.	.	.
23	10111	17	511	111111111	1FF
24	11000	18	512	1000000000	200
25	11001	19	.	.	.
26	11010	1A	.	.	.
27	11011	1B	.	.	.
28	11100	1C	1023	111111111	3FF
29	11101	1D	1024	1000000000	400
30	11110	1E	.	.	.
31	11111	1F	.	.	.
32	100000	20	.	.	.
33	100001	21	4095	111111111111	FFF
34	100010	22	4096	1000000000000	1000
35	100011	23	.	.	.
36	100100	24	.	.	.
37	100101	25	.	.	.
38	100110	26	65535	1111111111111111	FFFF
39	100111	27	65536	10000000000000000	10000
40	101000	28			

APPENDIX B: PINOUTS

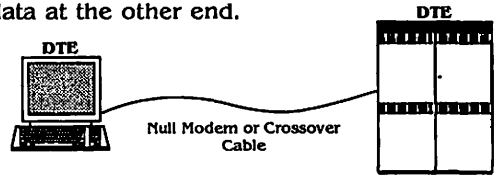
DTE/DCE

A few concepts that are often misunderstood in serial interfacing between computers are Data Terminal Equipment (DTE) and Data Communications Equipment (DCE). For the most part, computer serial ports, terminals and serial printers are DTE devices, while modems are DTE devices. Most of the problems encountered with new connections have to do with using the wrong cabling or connectors. The biggest difference between the devices is in their connector pin assignments. DTE devices are intended to talk to DCE devices as shown below. The connections between DTE and DCE devices are called "EIA extension cables", and allow straight through pin-for-pin wiring between devices.



The connections between DTE and DCE devices are called "EIA extension cables", and allow straight through pin-for-pin wiring between devices.

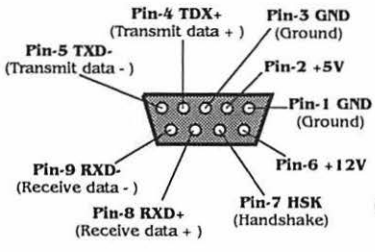
The problem with most direct terminal to computer (or computer to computer) connections is that both are usually DTE devices. DTE to DTE connections require a special cable that properly interconnects the signal lines. This type of cable is referred to as a "Null Modem Cable" or "Crossover Cable" because it crosses the data lines so that send data at one end becomes receive data at the other end.



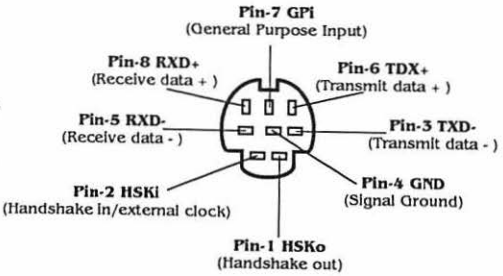
The rest of this appendix gives connection information for the communications ports found at the back of Macintosh computers.

The information given here should be used only as a guideline... please refer to Macintosh and VAX technical manuals for official technical information regarding connections to these ports. White Pine Software is not responsible for the accuracy of this information, or for any damage due to improperly connected equipment.

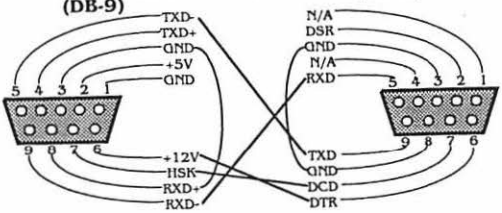
DB-9 Connector
Modem & Printer Ports



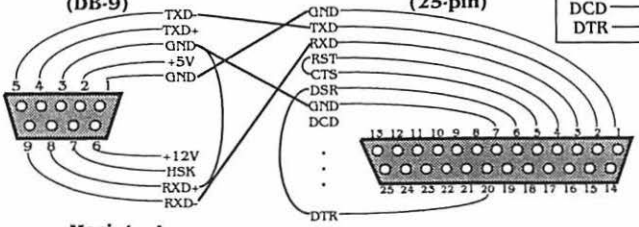
Mini-8 Connector
Modem & Printer Ports



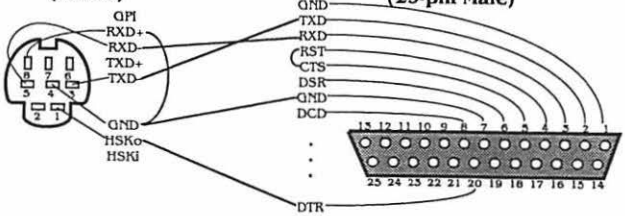
Macintosh Serial Connector (DB-9) to Apple Modem (DB-9)



Macintosh Serial Connector (DB-9) to Digital DF03 Hayes Smartmodem (25-pin)



Macintosh Serial Connector (Mini-8) to DEC Host Connection or Modem (25-pin Male)



Line Descriptions

GND	Ground
+5V, +12V	Power
GPi	General Purpose Input
HSK, HSKi, HSKo	Handshake Lines
TXD-, TXD+	Transmit Data Lines
RXD-, RXD+	Receive Data Lines
RST	Request to Send
CTS	Clear to Send
DSR	Data Set Ready
DCD	Data Carrier Detect
DTR	Data Terminal Ready

APPENDIX C: FOR MORE INFORMATION

This appendix gives details on obtaining additional information about topics related to file transfer. We've tried to include a broad range of sources for Macintosh to DEC VAX communications information, but this section is by no means conclusive. It's probably best to talk to the communications people of your organization for details to getting additional information about any of the concepts discussed in this booklet.

RELATED DOCUMENTATION

Visit the computer section of your local library or book store for more information on computer communications. Introductory computer science and most books on computer networking usually contain detailed discussions of most of the ideas and concepts given here. Trade journals and magazines are also a good place to pick up information on file transfers. A few of the more popular ones we know of are listed below.

Macworld
Subscriber Services,
P.O. Box 54529
Boulder, CO 80322-4529
800/642-9606

MacUser
P.O. Box 56986
Boulder, CO 80321-6986
(303) 447-9330

Digital News
P.O. Box 3
Winchester, MA
01890-9960
(617) 729-4200

DEC Professional
P.O. Box 503
Spring House, PA
19477-0503

Most major computer bulletin board systems (BBSs) also have helpful text files available for downloading, and many even have hotline areas, where BBS users can talk to communications experts, and message centers where information on communications hardware/software is posted. BBSs also usually contain lists of other BBSs in the local area. Local computer users groups or magazines also usually have BBS lists and information.

MACINTOSH NETWORKING

Below is a list of some of popular network protocol software currently available for the Macintosh.

DECNET

TSSnet
Alisa Systems, Inc
221 E. Walnut, Ste 175
Pasadena, CA 91101
(818) 792-9474

Community-Mac
Technology Concepts,Inc
40 Tall Pine Dr.
Sudbury, MA 01776
(508) 443-7311 x221
800-777-2323 x221

TCP/IP

TCPort/HostAccess
Kinetics, Inc
2540 Camino Diablo
Walnut Creek, CA 94596
(415) 947-0998

MacTCP
Apple Computer, Inc
20525 Mariani Avenue
Cupertino, CA 95014
(408) 996-1010

ADSP (AppleTalk Data Stream Protocol (see Apple Computer above for ADSP and AppleTalk Information))

FILE TRANSFER PROTOCOLS

XMODEM is a public domain program, and as such, is available, along with information and source code, on most computer bulletin board systems.

The **Kermit** protocol, source code, and information can be obtained by contacting:

Kermit Distribution
Columbia University
Center for Computing Activities
612 West 115th St.
New York, NY 10025

APPLE COMPUTER, INC

The manuals that came with your Macintosh are probably the best source of technical information about your system. However, if you need further information on Apple Macintosh computers and software products, contact your local Apple dealer.

DIGITAL EQUIPMENT CORPORATION

DEC has extensive documentation sets for its VAX systems, as well as networking and communications information. You should probably talk with your local VAX system manager if you have questions about your particular system. They should be able to show you the appropriate VAX manual to look through. You can probably also write to Digital Press for information on related texts, at the following address. (Digital Press, Digital Equipment Corporation, 12 Crosby Drive Bedford, MA 01730 1-800-343-8321)

WHITE PINE SOFTWARE, INC

For more information on Mac to VAX connectivity, or any of our products listed below, please feel free to contact us.

TERMINAL EMULATION

Mac220 - Text DEC VT220 terminal emulation
Mac240 - Text/Graphics DEC VT240 terminal emulation
Mac241 - Text/Graphics/Color DEC VT241 terminal emulation.

HOST BASED FILE TRANSFER

VMacS - VAX application for transfer and storage of Macintosh files on DEC VAX systems.

GRAPHICS CONVERSION

Reggie - Macintosh application for converting Mac graphics to DEC ReGIS & SIXEL formats.

X WINDOW SYSTEM

eXodus - Enables the Macintosh to function as a *Display Server* for any host system supporting MITs X Window System (A network based graphics & windowing system).

WHITE PINE SOFTWARE, INC
94 Route 101A, P.O. Box 1108
Amherst, N.H. 03031 U.S.A.

FAX# 1-603-886-9051
Telex 650-288-7627
AppleLink D0443

(603) 886-9050
9am to 5pm
(Eastern Time)

White Pine Software: The Leader In DEC/Apple Connectivity

White Pine Software was founded in 1984 by experts in system and application software dealing with communications and graphics on DEC and Apple computers. The company works closely with Digital Equipment Corporation and Apple Computer to insure full support of new computers, peripherals, and software. White Pine Software offers a complete line of DEC terminal emulation products, graphic conversion software and file transfer products. If you are not completely satisfied with any White Pine Software product, we offer a 30 day money back policy. Please consult your dealer for full details.

WHITE PINE SOFTWARE

94 Route 101A, P.O. Box 1108, Amherst, NH 03031 (603) 886-9050