Intermec
Technologies Corporation

Models 4920 and 4921 Series of PC Telecommunication Packages

REFERENCE GUIDE

PN: 961-021-013
Revision A
July 1995

" NOTICE

We welcome your comments concerning this publication. Although every effort has been made to keep it free of errors, some may occur. When reporting a specific problem, please describe it briefly and include the book title and part number, as well as the paragraph or figure number and the page number.

Send your comments to:
Intermec Technologies Corporation
Publications Department
550 Second Street SE
Cedar Rapids, IA 52401

This publication printed on recycled paper.

Acknowledgments

AS/400, IBM, IBM PC, Micro Channel, PS/2, and *OS/2* are registered trademarks and *AIX* and *Presentation Manager* are trademarks of International Business Machines Corporation.

Hayes is a registered trademark of Hayes Microcomputer Products Inc.

Metacomp is a trademark of Metacomp, Inc.

Some information in this guide has been reproduced courtesy of Metacomp, Inc.

APPENDIXES

APPENDIX K

FIGURES

TABLES

INDEX

Appendix A

Interpreting Log.Dat Information

The Models 4920 and 4921 Series of PC Telecommunication Packages generates a log file called LOG.DAT during each communication session. The file consists of various records that contain detailed statistics about the communication session.

If the “Restart communications” parameter in the system control file is set “No,” log information from the previous communication session is overwritten. If set “Yes,” log information from the restart communication session is appended to the end of the current log file, because the session is treated as a resumption of the interrupted communication session. You can change the setting for the current communication session by using the command line switch (for example, ncp4920/ry *or* ncp4920/rn). You can also enter “Yes” or “No” for “Restart communications” on the Main Menu.

Each record type output to LOG.DAT is identified by a 4-character log key, which is located at the beginning of the record. Log keys and their associated record types are defined as follows.

Use a text editor to view the contents of LOG.DAT, or create your own report.

" NOTE: *Hand-Held Computers will be referred to as HHC.*

Table A-1
LOG.DAT Contents

Log Key	Record Type
BEGA	activate begin record (autocall or autoanswer)
BEGD	498X communication session begin
BEGS	begin HHC communication session
BEGT	communication program begins
DDIR	498X device file directory entry
DVFS	498X device file update status
DVFU	498X device file update
EDAT	error download data file
EDNA	activate end record (autocall or autoanswer)
ENDS	HHC end-of-session status
ENDT	communication program end
FILE	download data file
LMSG	general log message

A log "level" from 0-3 is associated with each record type. A record is output to LOG.DAT only if its log level is less than or equal to the value set for the "loglevel" parameter in the SYSTEM.CTL file. The log levels and their associated log keys are as follows.

Table A-2
Log Key

BEGT, ENDT. Level 0 messages are always output to LOG.DAT, because the lowest value "loglevel" can be set to is 0.

LMSG, ENDS, BEGA, EDAT, DVFS, DDIR

BEGS, ENDA, BEGD, DVFU, (DVFU "abort" records, however, are *always* output to LOG.DAT.)

FILE

The abbreviation "498X" has been used throughout this appendix to indicate a 4980 Network Communications Controller or a 4985 Network Communications Controller.

Log keys and record types are defined on the following pages.

BEGT: The log key for the communication program begin record. Because its log level is 0, it is always output to LOG.DAT. The record contains the following fields:

Position	Length	Field
01	04	Log file key = BEGT.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
27	60	System version number message.

EXAMPLE:

```
BEGT 910803 163746     NCP4920 V 1.03 - begin communication
|---------|---------|---------|---------|---------|---------|--
1         10        20        30        40        50        60
```

BEGA: The log key for the activate (autocall or autoanswer begin record. It is written to LOG.DAT if the "loglevel" parameter in SYSTEM.CTL is set to "1" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = BEGA.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	4920 or 498X device channel identifier.

Position	Length	Field
23	02	4920 or 498X HHC channel identifier.
27	50	Activate parameters. Consists of a protocol type (2 = ADCCP, 3 = TTY, 4 = LAN) optionally followed by protocol-specific activate parameters. Each activate parameter consists of a dash (-), followed by a single character parameter type identifier and the parameter value. Activate parameters are described as follows:

TTY	ADCCP	LAN
-B <speed>	-B <speed>	-C <number of channels>
	-D <low address>	
-C <parity, data bits> 0 = none, 8 1 = odd, 7 2 = even, 7		
-A <flags> 8 = single tcom on	-E <high address>	
	-P <dial string>	
-P <dial string>		

EXAMPLE:

```
BEGA 910803 163801  00 02  2-PT3321
|---------|---------|---------|----
1         10        20        30
```

ENDA: The log key for the activate (autocall or autoanswer) record end. It is output to LOG.DAT when an activate record is deactivated and the "loglevel" parameter in SYSTEM.CTL is set to "2" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = ENDA.

06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	4920 or 498X device channel identifier.
23	02	4920 or 498X HHC channel identifier.
27	01	Error code. 1 = connection aborted, 2 = no answer, 3 = modem error, 4 = line disconnected, 5 = invalid parameter, 6 = general activate error, 7 = autoanswer timeout, 9 = line busy. Note that a line is normally disconnected after the last HHC exchange.
29	05	Activation count (number of times used).
35	05	"Good" terminal count for the activation period. The count is consistent with the "endil" and "endunsched" parameters set in SYSTEM.CTL.
41	50	Activate parameters. Consists of a protocol type (2 = ADCCP, 3 = TTY, 4 = LAN) optionally followed by protocol-specific activate parameters. Each activate parameter consists of a dash (-), followed by a single character parameter type identifier and the parameter value. See log key BEGA for activate parameters.

EXAMPLE:

```
ENDA 910803 163815  00 02  9 00001 00024 2-PT3321
|---------|---------|---------|----------|--------
1         10        20        30         40
```

BEGS: The log key for the begin HHC communication session record. The communication program writes a begin session record to LOG.DAT at the start of each HHC-to-host communication session if the "loglevel" parameter in SYSTEM.CTL is set to "2" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = BEGS.
06	06	Date in YYMMDD format.

Position	Length	Field
13	06	Time in HHMMSS, 24-hour clock format.
20	02	4920 or 498X device channel identifier.
23	02	4920 or 498X HHC channel identifier.
27	16	HHC identifier (HHC ID length is set in SYSTEM.CTL under the term "termidlen" parameter).
44	01	Session type. 0 = regular, 1 = initial download.
46	01	Added terminal flag. Y = yes, N = no.
48	05	Enabled request type list. 0 = regular download, 1 = initial download, 2 = clock, 3 = program, 4 = download status.

EXAMPLE:

```
BEGS 910803 163857  01 01  205                0 N 024
|---------|---------|---------|----------|---------|-
1          10         20         30         40         50
```

FILE: The log key for the download data file record. It is output to LOG.DAT whenever a file, clock data, or download status data is sent to an HHC, and the "loglevel" parameter in SYSTEM.CTL is set to "3." The record contains the following fields:

Position	Length	Field
01	04	Log file key = FILE.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	4920 or 498X device channel identifier.
23	02	4920 or 498X HHC channel identifier.
27	16	HHC identifier (HHC ID length is set in SYSTEM.CTL under the term "termidlen" parameter).
44	01	Request type. 0 = regular download, 1 = initial download, 2 = clock, 3 = program, 4 = download status.
46	40	File name.

EXAMPLE:

```
FILE 910803 163902  01 01  205              0 PACTM00.RAW
|---------|---------|---------|----------|---------|----
1         10        20        30         40        50
```

EDAT: The log key for the error download data file record. It is output to LOG.DAT when a problem occurs while a file is being processed and the "loglevel" parameter in SYSTEM.CTL is set to "1" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = EDAT.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	4920 or 498X device channel identifier.
23	02	4920 or 498X HHC channel identifier.
27	16	HHC identifier (HHC ID length is set in SYSTEM.CTL under the term "termidlen" parameter).

Position	Length	Field
44	04	Error number (see Section 9, "*Interpreting Error Messages*" in Volume B of the *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–012* for error numbers).
49	40	File name where error occurred.

EXAMPLE:

```
EDAT 910803 163950  01 01  201          0C00 PACTM00.RAW
|---------|---------|---------|----------|---------|----
1         10        20        30         40        50
```

LMSG: The log key for the general log message record. The communication program may write various informative or diagnostic messages to LOG.DAT if the "loglevel" parameter in SYSTEM.CTL is set to "1" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = LMSG.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
27	60	General message.

EXAMPLE:

```
LMSG 910803 163956          activate record load error
|---------|---------|---------|----------|---------|--
1         10        20        30         40        50
```

ENDS: The log key for the end-of-session status record. It is output to LOG.DAT if the "loglevel" parameter in SYSTEM.CTL is set to "1" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = ENDS.

06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	4920 or 498X device channel identifier.
23	02	4920 or 498X HHC channel identifier.
27	16	HHC identifier (HHC ID length is set in SYSTEM.CTL under the term "termidlen" parameter).
44	05	Upload record count.
50	05	Download record count.
56	01	Session type. 0 = regular, 1 = initial download.
58	01	"Added" flag. N = scheduled terminal, Y = added terminal.
60	01	Upload status. G = good, B = bad.
62	01	Download status. G = good, all data for HHC were sent successfully; B = bad, all data was not received by HHC.
64	01	Session status. G = good, HHC received all of its data; B = bad; W = warning, a data file error occurred on the PC host during the download session.
66	01	End-of-session error code. 0 = good, 1 = aborted, 4 = disconnected, 8 = HHC logic error (such as problem with download data).
67	04	HHC application dependent status. Refer to the HHC's documentation for the codes.

EXAMPLE:

```
ENDS 910803 163955  00 04  003           00024 00038 0 N
|---------|---------|---------|----------|---------|----
1         10        20        30         40        50
```

ENDT: The log key for the communication program end record. Because its log level is 0, it is always output to LOG.DAT. The record contains the following fields which correspond to the parameters on the "Statistics"

screen (see your *Communications with the Hand Held Computer* section 7 in Volume A in the *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–011):*

Position	Length	Field
01	04	Log file key = ENDT.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
27	05	Scheduled terminal count.
33	05	Number of good, scheduled, and regular terminals that communicated at least once.
39	05	Number of good, unscheduled, and regular terminals that communicated at least once.
45	05	Number of good, scheduled, and initial download terminals that communicated at least once.
51	05	Number of good, unscheduled, and initial download terminals that communicated at least once.
57	05	Number of warned and scheduled terminals that communicated at least once.
63	05	Number of warned and unscheduled terminals that communicated at least once.
69	05	Number of unsuccessful and scheduled terminals that communicated at least once.
75	05	Number of unsuccessful and unscheduled terminals that communicated at least once.
81	05	Total number of all successful and regular single and multiple sessions.
87	05	Total number of all successful and initial download single and multiple sessions.
93	05	Total number of all warned single and multiple sessions.
99	05	Total number of all unsuccessful single and multiple sessions.

" **NOTE:** *"Number of terminals" indicates the number of communication sessions with individual terminals, and does not include repeats for the same terminal. A terminal is scheduled if a record exists for it in the terminal database. "Number of sessions" includes repeats for the same terminal.*

EXAMPLE:

```
ENDT 910803 165854          00032 00001 00001 00000
|---------|---------|---------|----------|---------
1          10         20         30          40

 00000 00000 00000 00000 00000 00003 00000 00000 00000
|---------|---------|---------|---------|---------|---
50         60         70         80         90         100
```

There were 32 scheduled HHCs. One good, scheduled, and regular HHC communicated at least once. One good, unscheduled, and regular HHC communicated at least once. The total number was three good and regular single and multiple HHC exchanges.

BEGD: The log key for the begin 498X communication session begin record. The communication program writes a "begin device" record to LOG.DAT whenever a remote 498X controller starts a communication session and the "loglevel" parameter in SYSTEM.CTL is set to "2" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = BEGD.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	498X device channel identifier.
23	02	498X HHC channel identifier.
27	08	498X name. LOCAL = 4980 attached directly to a 4921 communications port.
36	05	498X software version.
42	40	Initialization parameters.

EXAMPLE:

```
BEGD 910903 234113  01 01  49850000 1.07  -S136
|---------|---------|---------|----------|-----
1         10        20        30         40
```

DDIR: The log key for the 498X device file directory entry record. The communication program writes a 498X directory record to LOG.DAT for each data directory entry uploaded from a remote 498X controller if the "loglevel" parameter in SYSTEM.CTL is set to "1" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = DDIR.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	498X device channel identifier.
23	02	498X HHC channel identifier.
27	08	498X name. LOCAL = 4980 attached directly to a 4921 communications port.
36	12	498X data file name.
49	06	YYMMDD file date stamp.
56	04	HHMM file time stamp.
61	06	File size in bytes.

EXAMPLE:

```
DDIR 910803 163950  00 01  49850000 BOOT.SYS    910301 1853
|---------|---------|---------|----------|---------|---------
1         10        20        30         40        50
```

DVFU: The log key for the 498X device file directory update record. The communication program writes a 498X file update record to LOG.DAT each time a file is created on a remote 498X controller and the "loglevel" parameter in SYSTEM.CTL is set to "2" or higher (note the following exceptions for aborted update records). The record contains the following fields:

Position	Length	Field
01	04	Log file key = DVFU.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	498X device channel identifier.
23	02	498X HHC channel identifier.
27	08	498X name. LOCAL = 4980 attached directly to a 4921 communications port.
36	01	Type of update. 2 = create or update, 3 = delete, 6 = abort (see explanation below).
38	40	Path name of the file on the host ("create" update type only).

An update is aborted if:

1. An error occurred while a previous update operation on a remote 498X controller was being processed. You can assume this error occurred if the DVFU record is preceded by a DVFS record with a non-zero status.
2. An error occurred while a file on the PC host was being opened or read.

Whenever an "abort" record occurs, the file system on the 498X is locked until the next host-to-498X session (the 498X will not boot HHCs). DVFU "abort" records are *always* written to LOG.DAT, regardless of the log level.

EXAMPLE:

```
DVFU 910803 163951  00 01     LOCAL 2 test/BOOT.SYS
|---------|---------|---------|----------|---------
1         10        20        30         40
```

DVFS: The log key for the 498X device file update status record. The communication program writes a 498X device file update record to LOG.DAT after the remote 498X controller sends the status of a file create or delete operation and the "loglevel" parameter in SYSTEM.CTL is set to "1" or higher. The record contains the following fields:

Position	Length	Field
01	04	Log file key = DVFS.
06	06	Date in YYMMDD format.
13	06	Time in HHMMSS, 24-hour clock format.
20	02	498X device channel identifier.
23	02	498X HHC channel identifier.
27	08	498X name. LOCAL = 4980 attached directly to a 4921 communications port.
36	01	Type of update. 2 = create or update, 3 = delete.
38	01	Update status code (see the following list).
40	12	File name on the remote 498X.
53	05	256-byte download record count (for "create" update type only).

The following list shows update status code.

Update Status

Code	Explanation
0	good
A	open error
B	read error
C	write error
D	maximum files
E	maximum open files
F	file not found
G	bad file handle
H	file in use
I	file system closed
J	space allocation
L	directory error

Code	Explanation
M	request aborted
O	file system locked

EXAMPLE:

```
DVFS 910803 163953 00 01   LOCAL 2 O BOOT.SYS      00020
|---------|---------|---------|----------|---------|-------
1         10        20        30         40        50
```

Appendix B

Understanding PL/N File Headers

Data is stored in the hand-held computer (HHC) in a form that minimizes the use of memory (RAM). The HHC's operating system and application program know how to store data keyed into the HHC; however, at communication time, the HHC must be told how to store data. Likewise, a host system cannot know the composition of the data it receives from the HHC unless it is told. During communication, it is the duty of the PL/N file header to reveal the structure of the data (which makes up the PL/N file) following it.

PL/N file headers are defined in the HHC application program. The details supplied in this appendix are only to help you understand PL/N files. You do not have to define them. When building them on your host, you will use information from your Program Specification. Remember that the Models 4920 and 4921 Series of PC Telecommunication Packages by Norand may hold multiple PL/N files.

Packing and unpacking data (also called transparent or binary data) are supported by the 4920 or 4921 packages. You can use this feature *only if your HHC application supports it.* Refer to your HHC Program Requirements Specification. Or, consult with your Norand Systems Engineer or Value Added Reseller (VAR) of NORAND® products.

Data Packing Terms

Before considering the different categories of PL/N file headers, you should know some terms used in data packing:

ASCII: a standard code that represents characters as binary numbers. This includes printable and control characters.

EXAMPLE: Linefeed key and carriage return key.

Binary Coded Decimal (BCD): a packed numeric, PL/N data type. A coding system in which each decimal digit is represented by a group of four binary 1s and 0s. It is used frequently to code numeric data and stores two decimal digits per byte.

Binary Byte: a packed numeric, PL/N data type. It is a 3-digit unsigned integer. This data type occupies one byte of memory.

Binary Word: a packed numeric, PL/N data type. It is a 5-digit unsigned integer. This data type occupies two bytes of memory.

8-bit, no parity is used with packed data.

Unpacked types are:

Log Key	Record Type
ASCII numeric	1 ASCII digit per byte
ASCII alphabetic	1 alphabetic character per byte
ASCII alpha-numeric	1 ASCII character per byte

PL/N File Header Format

Consider the following example of a typical header's format:

<DKPMSG 00024X001N006N004X032X001>

The header is made up of a number of parts, as follows:

Prefix "<" and Suffix ">" Characters: These two ASCII characters define the beginning and end, respectively, of the PL/N *file header.* The *data* that makes up the PL/N file header follows the ">" character.

File Type: This is the first position after the "<" character. The most common file types are:

B: Binary executable file. The data is stored in memory in a special way and can be executed as a "soft" PL/N program.

D: ASCII data file. This is considered HHC application program data (either uploaded or downloaded). It contains only ASCII characters.

E: ASCII executable file. A file containing ASCII characters only. The data is stored in memory in a special way and can be executed as a "soft" PL/N program.

P: Packed data file. A file containing transparent or "packed" data, which are interchangeable.

The preceding example is a data file, or type D.

File Name: The name always takes up six positions. If the name contains fewer than six characters, it must be left-justified and padded with ASCII spaces. The example shows file name KPMSG with one trailing space.

Record Count: The record count always takes up five positions. It is right-justified and padded with zeros. Under certain conditions, the record count may be all zeros. The example shows 24 records in the file.

Field Description: The field description tells the HHC how to store PL/N data in memory. The field description is made up of the field descriptor and the field length.

The field descriptor is always one character long. The most common data field descriptors are:

X: a general ASCII data field containing letters, numbers, and special characters. Each transmitted ASCII character takes up one byte of memory. Data is not packed.

N: a numeric binary coded decimal (BCD) data field. BCD data fields are either signed (+/-) or unsigned. Each transmitted ASCII character takes up half a byte of memory. Data in the HHC is packed into BCD format.

B: a binary byte data field. Some number of transmitted ASCII characters takes up one byte of memory. The number depends on the data field in the PL/N file header. Data is packed into a binary byte format.

W: a binary word data field. Some number of transmitted ASCII characters takes up two bytes of memory. The number depends on the data field in the PL/N file header. Data is packed into a binary word format.

A: an alphabetic ASCII data field containing letters only.

9: a numeric ASCII data field containing numbers only.

The field length, which follows each field descriptor, is always three decimal integers.

Record Description: The record description consists of one or more field descriptor and field length pairs. Each record in a PL/N file may consist of a number of data fields. The example shows five data field descriptors (X001, N006, N004, X032, X001) defined in PL/N file KPMSG.

PL/N File Categories

Fixed-Length File with Fixed-Length Records

The first category of PL/N files you may see always contains a nonzero record count in the header. When the PL/N file header is received during communication, the HHC knows immediately how much data follows. (The Models 4920 and 4921 Series of PC Telecommunication Packages do not use the header during communication.)

EXAMPLE: PL/N file header of this type, along with the data following it.

```
1    5   10   15   20   25   30   35   40   45   50   55   80
|----+---+----+----+----+----+----+----+----+----+----+...-|
<DTEST  00002X003N003N003N002>
A10123+1255
B11123-1266
```

1. This data file is named *TEST*. It consists of two records. Each transmitted record consists of 11 ASCII characters (3 + 3 + 3 + 2). The complete file, when communicated, consists of 52 ASCII characters (30 in the PL/N file header and 22 in the data).
2. Each data record consists of four fields: X003, N003, N003, and N002. (This is information used within the context of the HHC application program).

   The data in the first record is "A10123 + 1255."

   The first field, defined as X003, has the data "A10" stored in memory as three bytes.

   The next field, defined as N003, is defined within the HHC application program as an unsigned integer. The data "123" is stored in memory as two bytes, because memory is allocated in whole bytes.

The next field, defined as N003, is defined within the HHC application program as a signed integer. The data "+ 12" is stored in memory as two bytes.

The last field, defined as N002, is an unsigned integer. The data "55" is stored as one byte.

The data in the second record is "B11123 – 1266."

The first field, defined as X003, has the data "B11" stored in memory in three bytes.

The next field, defined as N003, is defined within the HHC application program as an unsigned integer. The data "123" is stored in memory as two bytes, because memory is allocated in whole bytes.

The next field in the record, defined as N003, is defined within the application program as a signed integer. The data "- 12" is stored in memory as two bytes.

The last field, defined as N002, is an unsigned integer. The data "66" is stored as one byte.

3. Each record in the file takes up eight bytes of memory. Both records together take up 16 bytes of memory.

Variable-Length File with Fixed-Length Records

The second category of PL/N files you may see always contains a zero record count in the header. The PL/N file header does not contain a record count. When it is transmitted, we need a special way of knowing when the file ends. An end-of-file delimiter, usually two ASCII tilde characters (~~), marks the end of the file and follows the data.

EXAMPLE: A typical file of this type:

```
1    5   10   15   20   25   30   35   40   45   50   55   80
|----+---+----+----+----+----+----+----+----+----+----+...-|
<DTEST  00000X003N003N003N002>
A10123+1255
B11123-1266
~~
```

1. The data in this file is interpreted and stored in memory the same as the fixed-length file with fixed-length records.

2. When you are creating data that goes into a PL/N file, sometimes it is more convenient not to keep a record count. If you count the records, you have to go back and update the record field in the PL/N file header. The alternative is to just write two tildes following the file data.

Variable-Length File with Variable-Length Records

The third category of PL/N files you may see always contains a zero record count in the header. Also, it only has one field descriptor, "X001," in the record descriptor area.

The PL/N file header does not contain a record count. When it is transmitted, we need a special way of knowing when the file ends. The end-of-file delimiter, usually two ASCII tildes (~~), commonly marks the end of the file and follows the data.

This type of PL/N file is used in HHC application programs to upload transaction details.

EXAMPLE: Details of selling or inventory management recorded during the day's activity.

The field descriptor included within the PL/N file header gives no clue to the structure of the HHC application data that follows. (Of course, the structure is known internally to the program.) The Models 4920 and 4921 Series of PC Telecommunication Packages knows its structures by referring to the UPLFMT.CTL file (see Section 14, *"Understanding the Upload Format Control File"* in Volume B of the *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–012*) when it detects this type of PL/N file header during data conversion (the "Format upload data" program).

EXAMPLE: A typical file of this type:

```
1    5   10   15   20   25   30   35   40   45   50   55   80
|----+---+----+----+----+----+----+----+----+----+----+...-|
<DBKTRXN00000X001>
A001
D100
C12345
E54321
```

```
Z050
~~
```

1. The file is named BKTRXN.
2. Before we can work with the file data, we need to know the file's structure. The UPLFMT.CTL file describes its variable-length records following this file's name.

The PL/N application with this file name in the upload format control file would reveal that this file has five record types, named A, D, C, E, and Z. The A record is three characters long. The D record is three characters long. The C record is five characters long. The E record is five characters long. The Z record is three characters long.

The data in the A record is "001." The data in the D record is "100." The data in the C record is "12345." The data in the E record is "54321." The data in the Z record id "050."

Arrays in PL/N File Headers

EXAMPLE: A PL/N file header with a structure resembling the following:

```
<DDRCTL 00001N005N008(004N006)000N004X016(006N006)000N004
X012X012>
```

1. This is a PL/N data file named DRCTL. It contains only one record (with a curious data field structure).
2. Array structure (004N006)000 follows the first two fields, N005 and N008. It defines four BCD fields, each having a length of six transmitted characters. The data fields they define are described in your HHC application program specification.
3. A second array structure, (006N006)000, appears in the header. This defines six BCD fields, each having a length of six transmitted characters.

Using Packed Data

The previous subsection described how the HHC stores data in a format that minimizes the use of memory. The Models 4920 and 4921 Series of PC Telecommunication Packages can pack PL/N data into this format, relieving the HHC of packing and unpacking.

The advantages of packed data include:

- The size of the files being transmitted is decreased. The HHC does not have to convert the data as it is received or transmitted.
- The net effect is shortened communication times and reduced phone costs.

You can use this feature only if your HHC application supports it. Refer to your HHC Program Requirements Specification, or consult with your Norand Systems Engineer.

Read the previous subsection on PL/N file headers to become familiar with type D and P files. To understand why processing needs to occur on the PC, you should understand the X, N, B, W, A, and 9 field descriptors within the PL/N file header.

You should also be thoroughly familiar with the UPLFMT.CTL file as described in Section 14, "*Understanding the Upload Format Control File*" in Volume B of your *User's Guide.*

Processing occurs during the download data preparation and upload data preparation phases of operation. Some specific examples follow.

Download Data Conversion

PL/N data in the SESSION.CTL file, HOST.DNL file, or a separate PL/N data file could be converted from ASCII data into binary data.

EXAMPLE: The following data. . .

```
<DDATA    0000X001N002>
A31
B32
C33
D34
E35
~~
```

would be converted to. . .

```
<PDATA    00000X001N002>A1B2C3D4E5~~
```

The file type has been changed from "D" (binary executable file) to "P" (packed data file). The type X fields remain in ASCII, but the type N fields changed from pairs of ASCII characters to one-byte BCD digits.

Upload Data Conversion

PL/N data in the upload file may be packed if the HHC is set up to upload packed data. An HHC can upload packed data if the "Pack PL/N data" parameter in the SYSTEM.CTL file is set "Yes." Any packed data in the upload file is unpacked.

EXAMPLE I:

The following data. . .

```
<PDATA    00000X001N002>A1B2C3D4E5~~
```

would be converted to. . .

```
<DDATA   00000X001N002>
A31
B32
C33
D34
E35
~~
```

A variable-length record file does not contain the unpacking information in the PL/N header, so the upload format control file must be used to unpack the data.

EXAMPLE II:

The following data. . .

```
<PUPTRXN000000X001>A1B05CMESSAGEx~~
```

using an upload format table containing. . .

```
file= UPTRXN
rectype=A
format=N002
rectype=B
format=N003
rectype=C
format=X006N002
```

would be converted to. . .

```
<DUPTRXN00000X001>
A31
B035
CMESSAGE78
```

Appendix C

Interpreting ADCCP Communication Errors

Communication errors displayed on your hand-held computer (HHC) fall into two categories: protocol and nonprotocol. The first step in troubleshooting a communication problem is to distinguish between these two types of errors.

A *protocol error* means there has been a violation of the rules Norand uses to transmit data between the HHC and another device (for example, phone line noise can abort communication before all data has been transmitted). The HHC's operating system reports these error types in a consistent manner.

A *nonprotocol error* means something is wrong with the data being transmitted. Generally, this would be a PL/N file header error or incorrectly formatted data in the PL/N file.

Most of these errors are also reported in a consistent manner; however, the application running in the HHC often determines exactly how the error should be displayed and reported back to the operator (this procedure is different for 4000 Series).

To fully use the information in this appendix, you must be trained by your Norand Systems Engineer in certain aspects of ADCCP protocol by Norand. You must understand how your NORAND® network is engineered and the part played by other system components.

If you are using equipment that is not from Norand, you must be able to tell if the problems are in it or the equipment by Norand. Furthermore, you must be able to tell if the phone lines, and not other parts of the system, are causing the problems.

This appendix is not meant to turn you into an instant troubleshooter of communication problems. However, if you understand some of the basics, if you isolate data and equipment problems, and if you replace components with temporary spares, you will solve your problems in less time and with minimal assistance.

" **NOTE:** *This appendix does not cover the error messages displayed on your NORAND multi-terminal controllers. These messages are described in the manuals that come with the controllers.*

Error Messages

Error messages are displayed on the HHC similar to the following example:

```
FILES RCVD: <number>
ERR: <error type and code>
```

Files Rcvd: is the number of files the HHC received from the host computer. The programmer puts the message into the HHC application. Your HHC may display a different message.

Err: is the error type and code. Error types and codes you may see are described as follows.

Error Type	Meaning
D	The HHC received PL/N file data that was incompatible with the corresponding PL/N file descriptor.
F	The HHC received data that could not be written to a PL/N file because memory is full, the file directory is full, or a "firm" file of the same name already exists.
H	The HHC received a PL/N file header that contained an invalid field descriptor.
L	Communication was aborted before the first PL/N file header was received.
T	The HHC experienced an unexpected end of transmission.

Error Type	Meaning
023	A PL/N file ended prematurely. (The PL/N file received did not contain a proper end-of-file delimiter.)
024	During formatting, the input variable did not agree in type with the data in the buffer.
026	The block of data that was received exceeded the expected number of bytes (128 or 256).
050	The HHC has received the TOKEN.
051	An ADCCP Disconnect command was received.
2XY	This code indicates a communication protocol error as described on the following table. A "2" means you are using an HHC with PL/N and ADCCP protocol. The "X" indicates the protocol phase in which the error occurred. The "Y" indicates the type of protocol error, which varies from 1 to 9, excluding 8.

Protocol Errors

Protocol Phase (X)		Protocol Error Type (Y)	
Value	Meaning	Value	Meaning
5	Transport layer TOKEN received	9	PUT buffer
4	Disconnect received	7	Clear state
3	In receiving state	6	Put state
2	In sending state	5	Get state
1	In inactive state	4	Internal
0	In a connectionless state (using modem)	3	Input timeout
		2	I/O tried after disconnect received
		1	Line lost (DSR, Clocks, CTS)

ADCCP Communication Error Example

```
FILES RCVD: 0
ERR: L 231
```

The message is FILES RCVD: 0. The HHC did not receive any files from the host because communication was aborted.

The error type is L and the error code is 231. The "L" means communication was aborted before the HHC received the first PL/N file header. The "2" indicates that your HHC was using PL/N and ADCCP protocol. The "3" indicates that the HHC was in the phase of receiving data from the host when the error occurred, and the "1" indicates a lost line. The program received the type and code from the operating system.

When you have a communication error, pay attention to the error type first. If it is "D," "F," or "H," you have PL/N file errors to look into. The protocol phase and error types are more easily determined after you eliminate PL/N file errors. Most of the time, an L0 or T0 error type indicates an application error.

The most common protocol phase values you will see are "3" (receiving) and "2" (sending). A "3" means the HHC is receiving data from the host. A "2" means it is sending data to the host.

The most common protocol error type is "1" (line lost). This refers to the RS-232 lines used to let the HHC know it is connected to the Models 4920 and 4921 Series of PC Telecommunication Packages by Norand through the Multidrop Quad Lockbox, or connected to an NM2400, NM2400A, or 9600 modem. Other error numbers are less common and should be brought to the attention of your Norand Systems Engineer, because the numbers very likely indicate a problem internal to the HHC application program.

Appendix D

Interpreting TTY Communication Errors

Communication errors displayed on your hand-held computer (HHC) fall into two categories: protocol and nonprotocol. The first step in troubleshooting a communication problem is to distinguish between these two types of errors.

A *protocol error* means there has been a violation of the rules Norand uses to transmit data between the HHC and another device (for example, phone line noise can abort communication before all data has been transmitted). The HHC's operating system reports these error types in a consistent manner.

A *nonprotocol error* means something is wrong with the data being transmitted. Generally, this would be a PL/N file header error or incorrectly-formatted data in the PL/N file.

Most of these errors are also reported in a consistent manner; however, the application running in the HHC often determines exactly how the error should be displayed and reported back to the operator (this procedure is different for 4000 Series).

To fully use the information in this appendix, you must be trained by your Norand Systems Engineer in certain aspects of the two-way TTY protocol by Norand. You must understand how your NORAND® network is engineered and the part played by other system components.

If you are using equipment from other companies, you must be able to tell if the problems are in it or the equipment by Norand. Furthermore, you must be able to tell if the phone lines, and not other parts of the system, are causing the problems.

This appendix is not meant to turn you into an instant troubleshooter of communication problems. However, if you understand some of the basics, if you isolate data and equipment problems, and if you replace components with temporary spares, you will solve your problems in less time and with minimal assistance.

" **NOTE:** *This appendix does not cover the error messages displayed on your NORAND multi-terminal controllers. These messages are described in the manuals that come with the controllers.*

Error Messages

EXAMPLE: Sample error messages displayed on the HHC:

```
FILES RCVD: <number>
ERR: <type and code>
```

Files Rcvd: is the number of files the HHC received from the host computer. The programmer puts the message into the HHC application. Your HHC may display a different message.

Err: is the error type and code. Error types and codes you may see are described as follows.

Error Type	Meaning
D	The HHC received PL/N file data that was incompatible with the corresponding PL/N file descriptor.
F	The HHC received data that could not be written to a PL/N file because memory is full, the file directory is full, or a "firm" file of the same name already exists.
H	The HHC received a PL/N file header that contained an invalid field descriptor.
L	Communication was aborted before the first PL/N file header was received.
T	The HHC experienced an unexpected end of transmission.
023	A PL/N file ended prematurely. (The PL/N file received did not contain a proper end-of-file delimiter.)

Error Type	Meaning
024	During formatted input, the input variable did not agree in type with the data in the buffer.
026	The block of data that was received exceeded the expected number of bytes (128 or 256).
1XY	This code indicates a communication protocol error as described on the following table. The "1" means you are using an HHC with PL/N and two-way TTY protocol. (Model 101 HHCs by Norand display error codes only. Also, they do not show the leading "1" in the error code.) The "X" indicates the protocol phase in which the error occurred. The "Y" indicates the type of protocol error, which varies from 0 to 9.

Protocol Errors

Protocol Phase (X)		Protocol Error Type (Y)	
Value	**Meaning**	**Value**	**Meaning**
5	Sign-on started	9	Control character error
4	Data send started	8	Block framing error: STX, ETX, LF, or CR
3	Turn-around started	7	Block check error
2	Data received started	6	Block count error
1	Sign-off started	5	Excessive NAKs
0	Filler	4	Data loss
		3	Character gap too long
		2	Parity error
		1	Line lost: DSR or CTS
		0	Data format error (101s only)

TTY Communication Error Example

FILES RCVD: 0
ERR: L104

The message is FILES RCVD: 0. The HHC did not receive any files from the host because communication was aborted.

The error type is L and the error code is 104. The “L” means communication was aborted before the HHC received the first PL/N file header. The “1” indicates that your HHC was using PL/N and TTY protocol. The “0” is a filler number (there is no protocol phase information to report), and the “4” indicates data loss. The program received the type and code from the operating system (this error can occur when you set up your HHC for communication and then remove the terminal from the lockbox or communication cable.)

Appendix E

Compatibility with Tcom Handlers

The Models 4920 and 4921 Series of PC Telecommunication Packages is designed to provide all the features of Tcom Handlers by Norand, but is not a direct replacement for them.

The communication needs of the 4000 Series of hand-held computers (HHCs) required significantly greater capabilities than Tcom Handlers could provide.

EXAMPLE: The need to communicate using TTY, ADCCP, NPCP (for local 4000 Series HHCs), and Ymodem (to cold boot 4000s) protocols.

You can use the 4920 or 4921 in a manner compatible with Tcom Handlers. Compatibility mode lets you provide data and control information to the PC in the same format used by Tcom Handlers and present upload data from the HHCs in the same format as the Tcom Handlers.

You can prepare data and control information, consisting of port activations and requests for data, on your host computer just as you did with the Tcom Handler. The format of the compatibility input file is identical with Tcom Handlers. However, the Tcom Handlers and the 4920 or 4921 take different steps to format the file after you have supplied it to the PC.

Similarly, the 4920 or 4921 can format upload data from the HHCs for use by your host computer just as Tcom Handlers did. You format the output into a single host file that has the same format as previous host upload files.

The file you use to prepare requests, activations, and data for the HHCs is called HOST.DNL. The program that formats that file for download is called HOSTDNF.EXE. You can find the instruction for preparing data with compatibility mode (Section 13 in Volume B of your *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–012.)* Section 5 in Volume A of the *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–011* tells you how to format the host download file for the HHCs.

Use the upload format program called HOSTUPF.EXE to prepare the UPLOAD.DAT file containing data from the HHCs for your host computer. Find instructions for this process in Section 6 in Volume A of your *User's Guide.*

Appendix F

Operating Unattended

The Main Menu is good to use when you are learning the concepts that underlie the Models 4920 and 4921 Series of PC Telecommunication Packages by Norand. But the Main Menu needs an operator to enter keystrokes. How do you go about customizing the system to run in an unattended mode? How can you get by with the least amount of human intervention? If you are going to fine-tune the system, you have to know some of the underlying pieces.

To get the most from the system and experience the fewest problems from your own modifications, you should:

- be familiar with your personal computer and its operating system
- understand the Models 4920 and 4921 Series of PC Telecommunication Packages

Acquaint yourself with the following programs and study the sample batch file in this appendix to see how to start out.

Table F-1
Executable Programs

MENU.EXE	Display the Main Menu
SESSFMT.EXE	Format the session control file
HOSTDNF.EXE	Format the host download file
PLNFMT.EXE	Format PL/N files for download
NCP4920.EXE	Start communications
HOSTUPF.EXE	Format the host upload file
UPLFMT.EXE	Format the upload file for the host
WAIT.EXE	Wait

Creating a Batch File

A batch file contains one or more commands the operating system executes consecutively. You can automate commands you use frequently by including them in your own batch file. The operating system executes them as if you had typed each one separately.

The batch file can be customized to replace the Models 4920 and 4921 Series of PC Telecommunication Packages menu system. If you want, you can shorten the batch file's name and type a single keystroke to start download format, communication with the HHCs, or upload format. Of course, the proper files must be in place for this operation to work. This means that MENU.CTL, SYSTEM.CTL, SESSION.CTL, COMM.CTL, and UPLFMT.CTL contain the correct information and are in the current directory.

This list provides general information about batch file names.

- File names can be in any combination of upper and lower case letters
- Batch files under DOS must have a .BAT file name extension. Batch files under OS/2 must have a .CMD file name extension
- Do not give a batch file the same name as an internal command, such as COPY.CMD or ERASE.BAT. Also, avoid having a batch file and a program with the same file name
- To carry out the instructions in the batch file, simply type the file name without its extension

Running Programs with Command Line Commands

This example shows you how to create a batch file that uses program names and switches. It formats data in the session control file, ends the communication session at a set time and date, and formats upload data. The commands in the batch file are the same as those you would use to start the communication session from the command line and format data using file names and switches.

To create a batch file, at the command line prompt type

```
C:\4920>copy con UNATTEND.<EXTENSION>
```

where <EXTENSION> is .BAT or .CMD, and press <Enter>. You can also use your own file name.

Now type the remarks and commands you want to include in the batch file. Press <Enter> after each line. When you finish, press F6 or Ctrl-Z and then <Enter>. The batch file is copied from your console and into the current directory.

EXAMPLE: Sample batch file

```
REM Do not display commands and remarks on the
REM screen.
echo off
REM
REM -Format the session control file-
REM
sessfmt SESSION.CTL /tn
REM
REM   -Communication with the HHCs-
REM
ncp4920 /t12:00 /d5
REM
REM        -Format upload data-
REM
hostupf UPLOAD.DAT HOST.UPL /cy
```

Type “unattend” to execute the instructions in this batch file.

Running Programs With MENU.CTL Commands

You can also create a file that formats data and starts the communication session by using commands stored in the menu control file (MENU.CTL) and switches. The commands in the file bypass the entire menu system.

The example shows you how to create a file that formats the session control file, ends the communication session at a set time and date, and formats upload data.

Remember that the system control file (SYSTEM.CTL) supplies default values for menu parameters. You can permanently change the default settings in the SYSTEM.CTL file, or use the "Edit the system control file" menu to change defaults.

To create this type of file, at the operating system prompt type

```
C:\4920>copy con MENU.INP
```

or use your own file name and extension, and press <Enter>.

Now type the commands you want to include in the file. Press <Enter> after each line. When you finish, press F6 or Ctrl-Z and then <Enter>. The file is copied from your console and into the current directory.

Here is a sample MENU.INP file:

```
s  SESSION.CTL /py
c  /t12:00 /d5
u  UPLOAD.DAT HOST.UPL /cy
```

Type "menu@menu.inp" to execute the instructions in this file.

Wait Utility

"Wait" is a utility you can use to run batched programs in a time window to control the duration of a pause between commands.

EXAMPLE: Formatting the session control file and starting the communication session.

Usually, you would place "wait" commands into a batch file. "Wait" also has a menu, which is discussed at the end of this appendix. You would most likely rarely use the menu.

Using Command Line Switches in Batch Files

For general information about batch files, see "Creating a Batch File" discussed earlier in this appendix.

Use "Wait" command line switches in a batch command stream to force a wait until a time of day is reached before the next command or program is executed.

To see what the command line switches for "Wait" are, at the operating system command line type:

C:\4920>wait /h

(where "h" means "help") and press <Enter>. The wait usage line appears:

wait: usage: wait [/thh:mm] [/dnn]

The following list explains usage line notation.

- wait = program name
- /thh:mm = wait end time, in 24-hour clock, HH:MM format
- /dnn = wait end date; "nn" the number of days from today
- required information

EXAMPLE I: Suppose batch file W.BAT was created and contains these commands:

```
wait /t15:00 /d3
ncp4920
```

This indicates that the waiting period ends at 3:00 PM three days from today, and then communication program NCP4920 begins. You would type "w" to execute the commands in this example.

EXAMPLE II: Suppose batch file GO.BAT was created and contains these commands:

```
wait /t11:30 /d00
ncp4920
```

This indicates that the waiting period will end at 11:30 AM zero days from today (such as today), and then communication program NCP4920 begins. You would type "go" to execute the commands in this example.

If the */d switch* is *omitted*, "Wait" will wait until the time on the PC equals the end time you indicated for the /t switch.

If you enter a time which is *less than* the current time, the waiting period ends on the following day at the time you indicated.

Wait Menu

The menu for "Wait" is not a task on the Main Menu. To retrieve the menu, type "wait" at the operating system command line prompt.

EXAMPLE:

```
Wait
<version> <date>
Copyright (c) Norand Corporation 1990

Current time:                  09:51
Current date:                09/11/91

Wait end time                  11:30
Wait end date:               09/14/91

     Wait      Help      Exit (ESC)

<message line>
```

Definitions

Current Time: Displays the PC's time in 24-hour clock, HH:MM format. If you need to change the time, return to the operating system command line prompt, type "time," and follow the instructions on the screen. See your operating system's reference manual for assistance.

Current Date: Displays the PC's date in MM/DD/YY format. If you need to change the date, return to the operating system command line prompt, type "date," and follow the instructions on the screen. See your operating system's reference manual for assistance.

Wait End Time: Sets the time at which you want the waiting period to end, in 24-hour clock, HH:MM format.

Wait End Date: Sets the date on which you want the waiting period to end, in MM/DD/YY format.

Wait: Sets the new wait end time and wait end date.

Exit (ESC): Exits the "Wait" program.

Procedure

Type the current end time or end date (or both) on the menu, then press <Enter> while the "Wait" command is selected on the menu. This sets the new wait end time and wait end date.

Appendix G

Booting 4000 Series HHCs

Due to improvements in hardware design in remote data communication technology, the 4000 Series hand-held computers (HHCs) maintain their operating system and programs in RAM (Random-Access Memory) rather than ROM (Read-Only Memory).

Previous generations of NORAND® HHCs retained their operating systems (and sometimes their programs) on EPROMs (Erasable Programmable Read Only Memory). This meant that when the HHC was powered off, it did not lose its operating system and programs (if they were also on EPROMs).

In contrast, our newest generation of HHCs, the 4000 Series, must have the operating system, programs, and (sometimes) data sent to them when you bring them into service for the first time. "Cold boot" is the industry jargon for bringing a computer to normal operating conditions when it has been without power. We have adopted this term to describe the same process for starting our 4000 Series computers.

This explanation makes clear the need for a convenient method for cold booting 4000 Series HHCs. The Models 4920 and 4921 Series of PC Telecommunication Packages meets that need. By placing the required files in the appropriate 4920 or 4921 directory, you can boot HHCs almost as easily as day-to-day communication with them.

Single 4000 Series Application

If your 4000 Series computers are running only one application program, just copy the application software into the boot directory named by the "bootdir" command in the system control file. At communication time you will not need to select an application, since the 4920 or 4921 package sends the only one available.

Multiple 4000 Series Applications

On the other hand, different 4000 Series computers may be running different application programs. If so, copy each application into the boot directory. The 4920 or 4921 package will not know which boot application to send during communication, however.

At communication time, you must select the boot menu from the menu bar on the "Communications status" screen, then highlight the boot application or applications you want the HHCs to receive. In the absence of a choice by you, the 4920 or 4921 sends the top HHC application appearing on the boot file list.

Loading Boot Sets

You must provide load records in the session control file for each application you want to download. The load (L) record names the boot set you want sent to HHCs.

Follow the load record (naming the boot set) with one or more file (F) records naming the files included in the boot set for the HHC application.

The L record and its associated F records identify files required to cold boot 4000 Series HHCs. An L record becomes current when it is encountered. All files specified in F records that follow L records are sent to the HHC when it requests a boot.

Name the first L record, then attach all F records for the first HHC application program immediately following it. Next, name a second L record and attach following that L record all F records for the second HHC application.

We also provide a "delete" switch for the load record. The /D switch on the load record has a different meaning depending on whether the mode record has been set to "new" or "update."

More details about the load record and its delete switch are provided in Section 12, "*Understanding the Session Control File*" in Volume B of the *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–012*.

Downloading Boot Sets

At communication time, the 4920's or 4921's communication program uses the L record to locate the appropriate F records for the application. It then prefixes the boot directory name to the program names you have specified with F records.

If you are viewing debug information, only the literal name specified by the L record appears, not the associated F record names.

You can find additional information about preparing to boot 4000s by reading Section 9, "*Understanding the Communication Control File*," Section 12, "Understanding the Session Control File," and the topic of boot directories (bootdir) in Section 10, "Understanding the System Control File" all in Volume B of your *User's Guide.*

Appendix H

File Processing

The Models 4920 and 4921 Series of PC Telecommunication Packages by Norand lets you maintain user files on remote 498X (4980 or 4985 Network Communication Controllers) devices.

To implement file maintenance for a device, follow these steps:

1. Enter an identifier into each 498X device which requires file maintenance. You can enter 4980 identifiers with the 4980 user interface. Enter 4985 identifiers with a set of 8 binary switches. (Refer to the appropriate 498X documentation for more information.)

" NOTE: *If a 4921 host system is connected directly to a 498X device, the device ID for the locally attached 498X device must be "LOCAL."*

2. Enter the device identifier in a device database on the 4920 host system. Enter a device by placing an N record in the session control file. The format of N records is explained below.
3. Enter a list of files for the device in a device file database on the 4920 host system. Enter device files by associating F records with a previous N record in the session control file. The format of F records is explained below.
4. Enable file processing on the 4920 system.

   To enable 498X file processing, use a text editor to add a FILES=Y parameter line to the COMM.CTL file immediately below the PORTS = parameter line. You can then edit it with the SYSTEM OPTIONS task on the "Edit the communications control file" menu.

Format of N Records

The format of an N record is:

N<device id> {STAtus=<0|4>] [INIt=<498X init parms>] [ZONe=<time zone offset>] [DIR=<Dnldir|Bootdir>] [TYPE=<B|N>]

<device id>: A 1 to 8 character 498X device identifier.

STAtus: 4 = set the real-time clock on the remote 4980 device (see ZONe). This is the default value.

1 = do not set the real-time clock on the remote device.

INIt: 498X specific initialization parameters as documented in the appropriate 498X user's guide. *Use caution when changing any of these parameters.*

ZONe: A value from -23 to 23, which specifies the time zone offset of a remote 4980 device in hours. You can use this parameter to set the real-time clock on a remote 4980.

" **NOTE:** *The 4985 does not have a real-time clock.*

If this parameter is nonzero, and STAtus equals 4, the time on the remote 4980 will be set to the time on the 4920 host and adjusted for the time zone offset.

DIR (D/B): If the Dnldir/Bootdir parameter has been specified in the system control file, you can use the DIR parameter to indicate that the files specified in trailing F records can be found in the Dnldir/Bootdir path.

" **NOTE:** *The Dnldir/Bootdir path name is simply prefixed to the path name specified in the F record.*

TYPE: Use the TYPE parameter to indicate that the files specified in trailing F records are or are not 4000 Series hand-held computer (HHC) boot files. The default type is "N" (none). See "*Booting 4000 Series HHCs*" at the end of this appendix.

Format of F Records

The format of an F record is:

F<file path name>
[DIR=<Dnldir|Bootdir|none>]
[STAtus=<0|1|2>] [TYPE = <B|N>]

<file path name>: the full DOS or OS/2 path name of the file, unless:

- the file is in the working directory, or
- the Dnldir or Bootdir parameter was specified for the file.

DIR (D/B): If the Dnldir/Bootdir parameter has been specified in the system control file, you can use the DIR parameter to indicate that the file specified in the F record can be found in the Dnldir/Bootdir path. The DIR parameter on an F record overrides the DIR parameter on an associated N record.

" **NOTE:** *The Dnldir/Bootdir path name is simply prefixed to the path name specified in the F record.*

STAtus: 0 = enable file
1 = disable updates for this file
2 = force an update for this file

TYPE: Use the TYPE parameter to indicate that the file specified in the F record is or is not a 4000 Series HHC boot file. The TYPE parameter on an F record overrides the TYPE parameter on an associated N record. See "Booting 4000 Series HHCs" at the end of this appendix.

General Operation

The 4920 or 4921 maintains the list of files associated with a device, based on the following criteria:

1. The file name, time, and date for each file are entered into a directory on the remote 498X device.
2. At the start of each device session, the 498X file directory is uploaded to the 4920 or 4921 system.
3. Files which are in both the maintenance list for the device and in the 498X directory, and have matching time and date stamps are not updated.
4. Files which are in both the maintenance list but not in the 498X directory, or which have different time and date stamps, are downloaded to the 498X. The file has to be *enabled.*

5. If an *enabled* file is listed in the 498X directory but is not on the 4920 or 4921 list of files for the device, the file will be deleted on the 498X.

" **NOTE:** *You can prevent files from being deleted by creating a* disabled *file entry for the file in the maintenance list for the device.*

If file processing is enabled on the 4920 or 4921 system, then file processing is enabled on each remote device. Refer to the appropriate 498X programmer's guide for specific information about how file processing is implemented on a 498X device.

The 4920 or 4921 host "aborts" file processing if a 498X file update operation fails. If file processing is aborted on both the 4980 (versions 1.07 and later) and the 4985, the file system on the 498X locks for HHC access (i.e., the HHCs in a dock attached to the 498X will not be able to boot). The 498X file system unlocks as soon as the host completes successfully an entire file processing session. The 498X device also lock its file system during the file update process.

After the file maintenance list has been defined for a device, you can update a file by simply overlaying the old file with the new file, on the 4920 or 4921 host.

Booting 4000 Series HHCs

Files specified as boot files (with the TYPE = boot parameter) are entered into a boot list file for the associated 498X device.

When session control formatting is executed, the boot list file is created in the working directory with the device ID as the file name and $PL as the file extension. The boot list file is sent to the 498X as NETRPL.LST.

The boot list file consists of 11-byte records, with each record containing an 8-character file name (padded with blanks) and a 3-character file extension (padded with blanks). The file must be terminated with a blank record.

" **NOTE:** *The working directory for the 4920 or 4921 communications prgoram must be the same as the working directory for session control formatting, so that the boot list files can be found during communication.*

Appendix J

4000 Series Warm Start System

You can implement an efficient 4000 Series hand-held computer (HHC) warm start system. The warm start system lets you distribute new versions of HHC programs by copying them into a Model 4920 or 4921 directory. The program files can be stored on remote 4980 or 4985 NORAND® Network Controllers and obtained locally by 4000 Series HHCs.

Warm start capability is available beginning with the Models 4920 and 4921 Series of PC Telecommunication Packages version 1.07, 4980 version 1.10, and 4985 version 1.00.

To use the Model 4920 or 4921 and 4980 or 4985 warm start systems, you must change the HHC application program. The change has two parts:

1. implementing program updates
2. implementing DWNLRQ file requests

The first change requires the application to overlay itself with an updated version. The second change requires it to interpret a command file. This file directs the HHC to set up for a second session, and then request a program update from a 4980 or 4985 DWNLRQ file server or a DWNLRQ server on the host. The Model 4920 or 4921 provides both facilities for consistency.

The HHC application binds to the 4980 or 4985 file server called NORAND_SERVER. The HHC application should be coded to obtain new program files from the host when the files are *not* available from the 4980 or 4985 server (because the HHC is attached to a modem and is using TTY to communicate).

" **NOTE:** *The receive routine IPFRCV commonly used on a 4000 Series HHC cannot pack 7-bit Z-space hex files. Therefore, any PL/N application that implements automatic program updates must use the 8-bit binary format.*

Implementing the System

To implement a Model 4920 or 4921 warm start, you should follow these steps. *Each is discussed in greater detail in the following pages.*

1. Set the identification (ID) on remote 4980 or 4985 controllers.
2. Use the Model 4920 or 4921 SESSION.CTL file interface to create a file maintenance list for each remote 4980 or 4985 controller ID.
3. Enable file processing.
4. Understand DWNLRQ file servers.
5. Create an "command file" and add a program load request for the command file for each HHC in the Model 4920 or 4921 SESSION.CTL file. A single request can be broadcast to all HHCs.
6. Set the current version number of the HHC application.
7. Update the 4920 or 4921 system with a new application when appropriate.
8. Convert the HHC program to support program updates and DWNLRQ program requests.

For additional information about record types and parameters, refer to Section 10, "Understanding the System Control File," and Section 12, "Understanding the Session Control File" in Volume B of your *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–012.*

Setting the Controller ID

The 4980 default ID is NORAND. The 4985 default ID is 49850000. A unique ID is *not* required for each controller.

Creating a 4980 or 4985 File Maintenance List

The 4980 or 4985 file maintenance system lets you create, update, and delete files stored in the controller's random-access memory (RAM).

The 4985 RAM file system is nonvolatile for at least one hour. The 4980 file system is volatile. You can save files to disks with the 4980 front panel user interface by setting FILE PROC? to "Yes." The 4980 or 4985 file maintenance interface is documented in the controller's programmer's guides.

Create a file maintenance list for each 4980 and 4985 controller. A file list is associated with a 4980 or 4985 ID in the Model 4920 or 4921 SESSION.CTL file.

" **NOTE:** *The ID of a 4980 or 4985, which is directly cabled to a 4921 host, is always "LOCAL."*

An example of a Model 4920 or 4921 SESSION.CTL maintenance list follows. "N" is the designation for a Network Communications Controller Identifier record. "F" is the designation for a File Identifier record.

```
N 49850000 TYPE=BOOT DIR=BOOTDIR
F PACSM0P TYPE=DATA DIR=DNLDIR
F PACSL0P TYPE=DATA DIR=DNLDIR
F BOOTPH0.SYS
F SYSARC.EXE
```

In this example:

- The controller ID is 49850000.
- The TYPE and DIR parameters in the N record set the default type to BOOT, and the directory to BOOTDIR, for the files that follow.
- The BOOT type parameter generates for the controller a boot list called NETRPL.LST.
- The DIR parameter makes the Model 4920 or 4921 software search the directory specified.

" **NOTE:** *BOOTDIR and DNLDIR are Model 4920 SYSTEM.CTL file parameters. The SYSTEM.CTL file specifies the actual path names.*

The TYPE and DIR parameters in the first two F records override the parameters in the N record. Files listed in F records are maintained on each 4980 or 4985 controller with the same ID number. More information on N and F records is provided in Appendix H, *"File Processing."*

Files in the maintenance list download to controllers with ID 49850000 when the time and date stamp for the file on the controller do *not* match the time and date stamp for the file on the PC host. Files *not* in the list are deleted from the 4980 or 4985 file system. An F record "status" parameter can prevent files from being deleted from a 4980 file system.

A correlation exists between the files in the above example and booting or warm starting a 4000 Series terminal.

- The BOOTPH0.SYS file is required for Ymodem cold starts, and can be ignored.
- The PACX files (PACXM0P, PACXL0P) represent modules in the HHC PL/N application.
- The SYSARC.EXE file is a self-extracting archive that contains the HHC operating system files and a "kernel" program.

An HHC receives the SYSARC file when the HHC is cold started. After the operating system boots, it passes control to the kernel program. The kernel program immediately initiates a session with the DWNLRQ file server to obtain the HHC application modules. The boot process is complete when control passes to the application.

After the application is running, it can overlay (that is, warm start) itself by passing control to a communications module. The communications module requests the new PACX files from the server. The Model 4920 or 4921 host can initiate the warm start by scheduling a program load request for the HHC (see "*Create a Program Command File*" later in this appendix).

" NOTE: *The 4980 or 4985 file maintenance system is not restricted to boot and warm start files. Refer to the appropriate product documentation for additional information.*

The 4980 has up to 370K of space available for user data files, depending on the 4980 port configurations. The 4985 has 340K of space available for user data files.

Enable File Processing

To enable file processing, set the "File Processing" parameter on the Model 4920 or 4921 "Edit the communications control file" menu to

"Yes." You can also change the "Files = " default parameter in the COMM.CTL file to "Y."

Understanding DWNLRQ File Servers

The Model 4920 or 4921 and 4980 or 4985 DWNLRQ file servers enable 4000 Series HHCs to request files from a locally attached NORAND LAN controller. The session interface is identical with the interface 4000 Series terminals use to communicate with a host computer. To access the server, the application must change the name of the target application to which it is binding from NORAND_HOST to NORAND_SERVER.

The server expects file requests to be in the form of a PL/N DWNLRQ request file.

EXAMPLE:

```
<DDWNLRQ00003X016>MYDATA.DAT    PMBNXOP    PMBNGOP
```

The server sends the files as requested, until:

- a fatal communications error occurs,
- a file cannot be found, or
- all files have been sent.

The server supports MS DOS file names and extensions, but does *not* support a path name. The Model 4920 or 4921 host software also supports DWNLRQ requests. This indicates that terminals (that is, an NT141XL, NT141GL, or 4000 Series HHC) can consistently request files from the host if the LAN server fails or the terminal is *not* attached to a NORAND LAN. All Model 4920 and 4921 DWNLRQ files must be in the DNLDIR directory specified in the Model 4920 or 4921 SYSTEM.CTL file.

Create a Program Command File

The Models 4920 and 4921 updates HHC application programs. A program version number and program load requests are associated with HHC identifiers in the Model 4920 or 4921 SESSION.CTL file. The

load requests are triggered whenever the program version on the HHC does *not* match the program version associated with the HHC identifier on the PC host. Depending on the HHC application, this facility can be used with or without 4980 or 4985 file maintenance and DWNLRQ server support.

The HHC application must be programmed to overlay itself. If the HHC application *does not support* the DWNLRQ file server, direct the Model 4920 or 4921 communications software to download a new program to each HHC that has an out-of-date application.

If the HHC application *does support* the DWNLRQ file server, direct the Model 4920 or 4921 software to download a command file to the HHC. The command file directs the HHC to set up for a second session and receive the set of program files specified in the command file.

If the HHC is on the NORAND LAN, the HHC can request the files directly from the local LAN controller (with the advantage that the program files would only be sent over the phone lines once). Otherwise, the HHC must request the files from the host. The Model 4920 or 4921 host similarly supports non-LAN DWNLRQ requests.

" **NOTE:** *The Model 4920 or 4921 and 4980 or 4985 software do not dictate the format of the command file.*

One way to change the format is to make the command file look exactly like a DWNLRQ request file. Converting from program downloads to command file downloads requires changing the broadcast program load requests in the Model 4920 or 4921 SESSION.CTL file. The following examples show how to implement automatic program updates with the Model 4920 or 4921 SESSION.CTL file interface.

EXAMPLE I:

```
M VERSION=1.01
B PACXM0P TYPE=3
B PACXL0P TYPE=3
T 100
T 101
```

Example 1 assumes the HHC application does *not* support the DWNLRQ file server facility. The host computer sends the entire application with the PACX modules to each terminal (for example, 100

and 101) whenever the version number in the HHC session control record does *not* match the global version number the M record specifies (1.01 in the example). The Model 4920 or 4921 SYSTEM.CTL file specifies the position and length of the version string in the HHC session control record.

EXAMPLE II:

```
M VERSION=1.01
B PACX.CMD TYPE=3
T 100
T 101
```

Example 2 shows how the SESSION.CTL records in Example 1 would be changed if the HHC application *supported* the DWNLRQ file server facility. PACX.CMD is a command file that directs the HHC application to bind locally with the 4980 or 4985 DWNLRQ server to obtain the files specified in the command file.

" **NOTE:** *This example assumes that the HHC files are maintained on remote controllers as described in the preceding example.*

Setting the Version Number

Set the current version number of the HHC application. You can enter a version number in M or T records in the Model 4920 or 4921 SESSION.CTL file. A version number on the M record is global, whereas a version number on the T record applies to a specific terminal. The T option is provided for multiple application support.

Updating with a New Application

To update the 4920 or 4921 system with a new application, copy the HHC application program files into the Model 4920 or 4921 DNLDIR directory. Copy the HHC system files into the Model 4920 or 4921 BOOTDIR directory.

After you have installed the warm start, you can distribute a new application by copying the HHC application files into the DNLDIR directory.

Converting HHC Application Programs

Convert the HHC program to support program updates and DWNLRQ program requests.

Once the warm start system is in place, you can distribute a new application by copying the program files into the host download directory.

Due to hardware design improvements in remote data communications technology, the 4000 Series HHCs maintain their operating system and programs in RAM (Random Access Memory) rather than ROM (Real-Only Memory). Therefore, the 4000 Series HHCs must have the operating system, programs, and information sent to them when you bring them into service for the first time (cold start).

If your 4000 Series HHC is running only one application program, copy the application software into the boot directory named by the BOOTDIR command in the SYSTEM.CTL file.

Program Support

You would normally order 4980 system diskettes with HHC system files and application programs in a single archive file. However, if you are using the warm start system, you should order the 4980 system diskettes with no HHC files or with an archive file that contains the HHC system files and a “kernel” application. The kernel is a small application that requests the HHC application. The HHC application program files should be maintained on the 4980 from the PC host.

" **NOTE:** *A kernel can be replaced by a duplicate boot copy of the application if space is available.*

If the network includes 4985 controllers, the Models 4920 or 4921 PC host should maintain all of the HHC files (system files, kernel, and application).

Appendix K

4920 Telecommunication Network

4920 Multi-tasking Overview

The Models 4920 and 4921 Series of PC Telecommunication Packages have been the backbone for NORAND® Telecommunication Networks since 1989. Norand continues to support and enhance the current product to meet new marketing trends and customer business practices. Customers in the past have been using the Norand system on networks or with Host Emulation Software running in the background, many of these customer have been either limited in what they could perform or have to reboot between sessions because of 492x memory constraints. Norand has, and will continue to maintain that the 492x products function most efficiently on dedicated systems. However Norand is introducing “4920 Lite,” which will offer customers much more memory to run background tasks.

4920 software program loads its drivers onto the Metacomp board through a shared memory segment located in the extended memory region. This memory segment is declared in the MPLD.CTL file (Factory Default: D000).

Network Configurations

Installation Notes

Most network cards are installed at the same memory address as the default 4920 memory card (D000 - located in the 4920 MPLD.CTL file). The 4920 requires a full 64K memory region to function properly whereas the network cards normally require approximately 16K. It

may be more advantageous to your system configuration to move the network card to a new region rather than find a free 64K contiguous memory segment to install the 4920 Metacomp Card. A conflict in memory addressing with the 4920 Card and network cards normally results in a *1501 Communication Port Open Error* when attempting to start the 4920 communication program.

Networking Options

The 4920 Telecommunication Package does function on many network operating systems. There are performance considerations that will enhance your overall network performance. A number of examples are given below to illustrate the 4920 functionality on a network, however these important factors should be used in planning your implementation:

1. Upload and download file access. Communications is a very intensive file access process. Accessing files across the network should be minimized to increase your overall performance.
2. If network downtime is an issue or you wish to guard against the HHCs being affected by network downtime, actual communication files should be located on the local PC, I/O files (HOST.DNL, SESSION.CTL, or HOST.UPL) can be written or read from the network drive to provide automated access to "back end" applications.
3. Is concurrent upload processing needed or planned for the future? If YES, then examine closely the implementations listed in the next three examples, download data should be on the local PC, upload data *can* be redirected to the network drive.
4. Are more than one communication system being used on the same network (for example: access to common download data is needed on separate PCs), follow the next three examples which discusses the implementation.

EXAMPLE I:

Situation: 4920 or 4921 using download and upload files located on the network drive.

Recommendation:

1. Using 492x programs: HOSTDNF.EXE, SESSFMT.EXE, PLNFMT.EXE, HOSTUPF.EXE, or UPLFMT.EXE point to or from network drive to transfer files between local and network drives.

2. All 4920 software is loaded on local PC.
3. All 4920 directories and paths in SYSTEM.CTL file should point to the local PC.
4. Network software can be loaded as a TSR and all virtual drives mounted.

Advantages:

1. Local access to upload or download files for quicker access and better efficiency.
2. Decreases network activity.
3. Provides full backup and access to upload or download by HHCs even if the network is down.

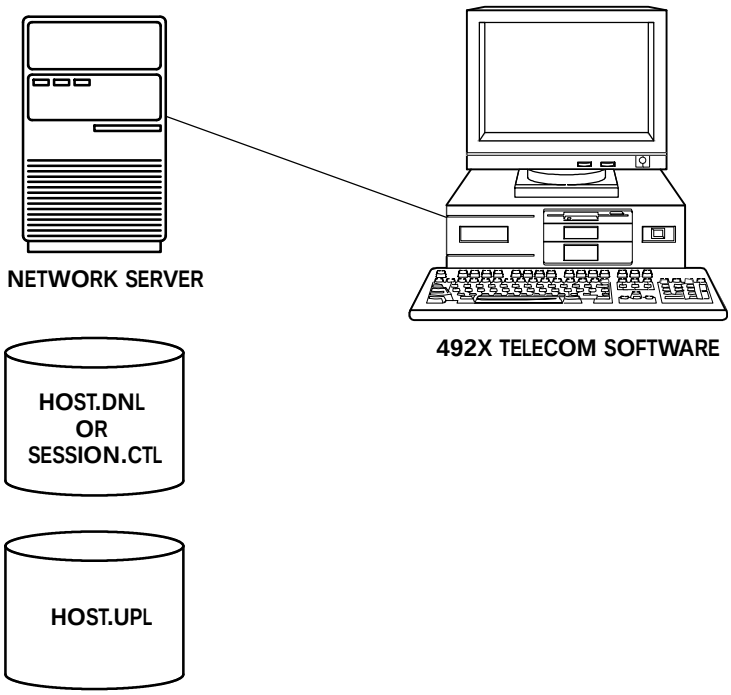

Figure K-1
Example 1

" *4920 Software installed on local PC*

- *Network Software installed*
- *Minimum 520K Free RAM (See Table K-1)*
- *4920 Software Download Formatters need to point to network drive for file transfer.*

  PLNFMT N:\HHCDATA\INCLUDE.DAT C:\4920\DOWNLOAD\INCLUDE.INC. Examples:

  HOSTDNF N:\HHCDATA\HOST.DNL
  SESSFMT N:\HHCDATA\SESSION.CTL
- *4920 Software Upload Formatters need to point to network drive for file transfer. Examples:*

  HOSTUPF C:\4920\UPLOAD.DAT N:\HHCDATA\HOST.UPL
  UPLFMT C:\4920\UPLOAD.DAT
- **** SYSTEM.CTLs UPLDIR should be set to N:\HHCDATA\UPLOAD***

EXAMPLE II: **Situation:** 4920 or 4921 using download files on local PC and upload files written to the network drive.

Recommendation:

1. Using 492x programs: HOSTDNF.EXE, SESSFMT.EXE, PLNFMT.EXE, point to or from network drive to transfer files between local and network drives.
2. All 4920 software is loaded on local PC.
3. All 4920 directories and paths in SYSTEM.CTL file should point to the local PC, except UPLOAD.DIR can point to network drive if using concurrent upload formatting.
4. Network Software can be loaded as a TSR and all virtual drives mounted.
5. Enhancement added in V1.14 of 4920 software allows for renaming and writing of the "raw" upload file to the network drive. This parameter has to be added to the SYSTEM.CTL file with a text editor, add the following line: UPLOADFNAME=N:\HHCDATA\UPLOAD.DAT.
6. Critical files to the 4920 operation, executables, device drivers (LOAD directory), database files (*.idx and *.dat) files, should all be located on the local PC.

Advantages:

1. Local access to download files provide for better efficiency.
2. Decreases network activity.

3. Provides “raw or unformatted” upload data onto network drive, for back-end upload processing and advanced 492x concurrent formatting of upload.

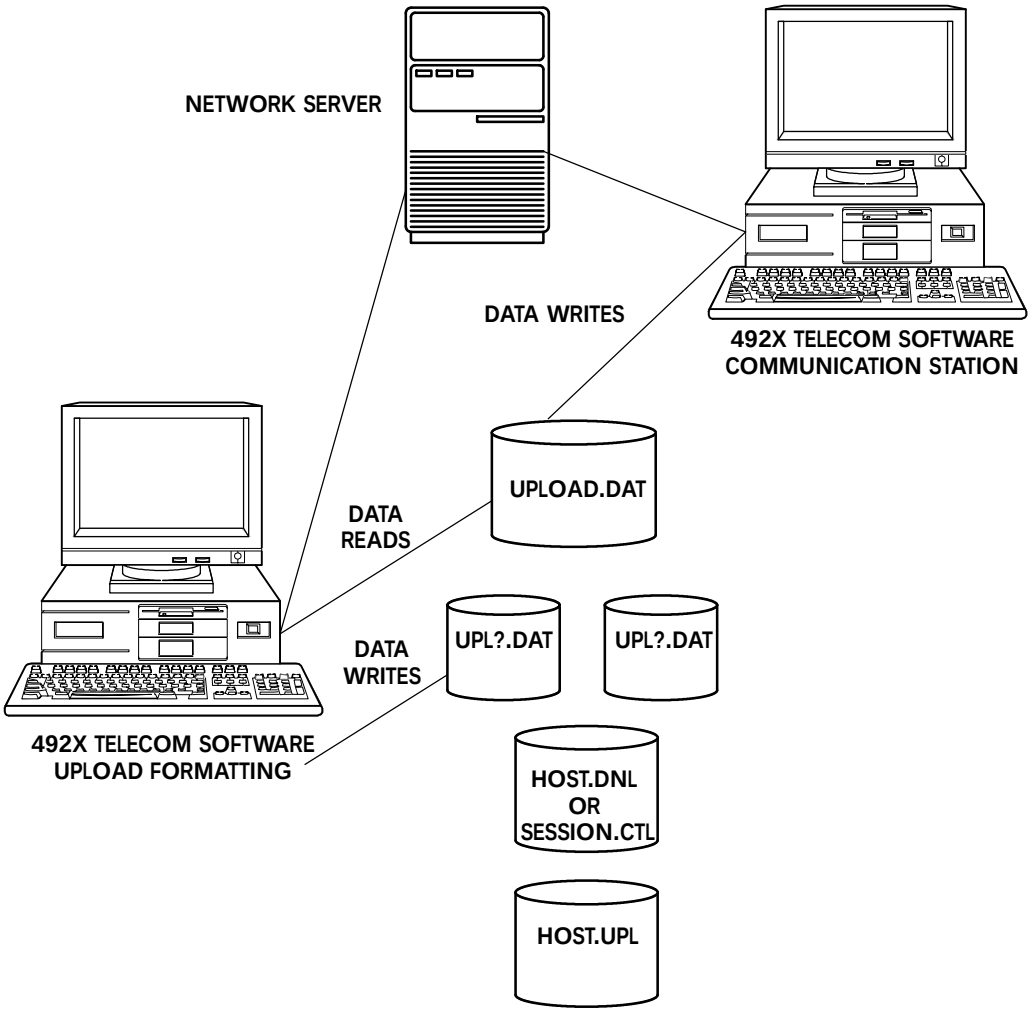

Figure K-2
Example 2

" **NOTE:** *Review 4920 and 4921 User's Guide NPN: 961–021–011 for concurrent upload formatting description and usage.*

- *4920 software installed on local PC*
- *Network software installed*
- *Minimum 520K free RAM (See Table K-1)*
- *4920 software download formatters need to point to network drive for file transfer. Examples:*

  PLNFMT N:\HHCDATA\INCLUDE.DAT C:\4920\DOWNLOAD\INCLUDE.INC
  HOSTDNF N:\HHCDATA\HOST.DNL
  SESSFMT N:\HHCDATA\SESSION.CTL
- *4920 software upload formatters can pull the files in from the network or a second PC on the network can be installed with the 4920 and processing of upload concurrently by accessing the "raw" upload file on the network drive. Examples:*

  HOSTUPF N:\HHCDATA\UPLOAD.DAT N:\HHCDATA\HOST.UPL
  UPLFMT N:\HHCDATA\UPLOAD.DAT
- *SYSTEM.CTLs UPLDIR should be set to N:\HHCDATA*
  add to SYSTEM.CTL- UPLOADFNAME=N:\HHCDATA\UPLOAD.DAT
- *Customers who use more than one communication station can also send a list of upload files to the upload formatter.*

EXAMPLE III: **Situation:** 4920 or 4921 using download and upload files on network drive.

Recommendation:

1. Using 492x programs: HOSTDNF.EXE, SESSFMT.EXE, PLNFMT.EXE, point to or from network drive to process download and upload files.
2. All 4920 software is loaded on local PC.
3. 4920 directories and paths in SYSTEM.CTL file should point to the network drive, except LOAD.DIR should point to local drive.
4. Network Software can be loaded as a TSR and all virtual drives mounted.
5. Enhancement added in V1.14 of 4920 software allows for renaming and writing of the "raw" upload file to the network drive. This parameter has to be added to the SYSTEM.CTL file with a text editor, add the following line: UPLOADFNAME=N:\HHCDATA\UPLOAD.DAT.

Advantages:

1. All upload and download files are contained on the network for access by back end software or other communication boxes.
2. Provides "raw or unformatted" upload data onto network drive, for back-end upload processing and advanced 492x concurrent formatting of upload.

Disadvantages:

1. Heavy network traffic by reads from download files and writes of upload files.
2. Network activity or load could affect 4920 communication program function.
3. Decreased communication efficiency and increased phone costs.

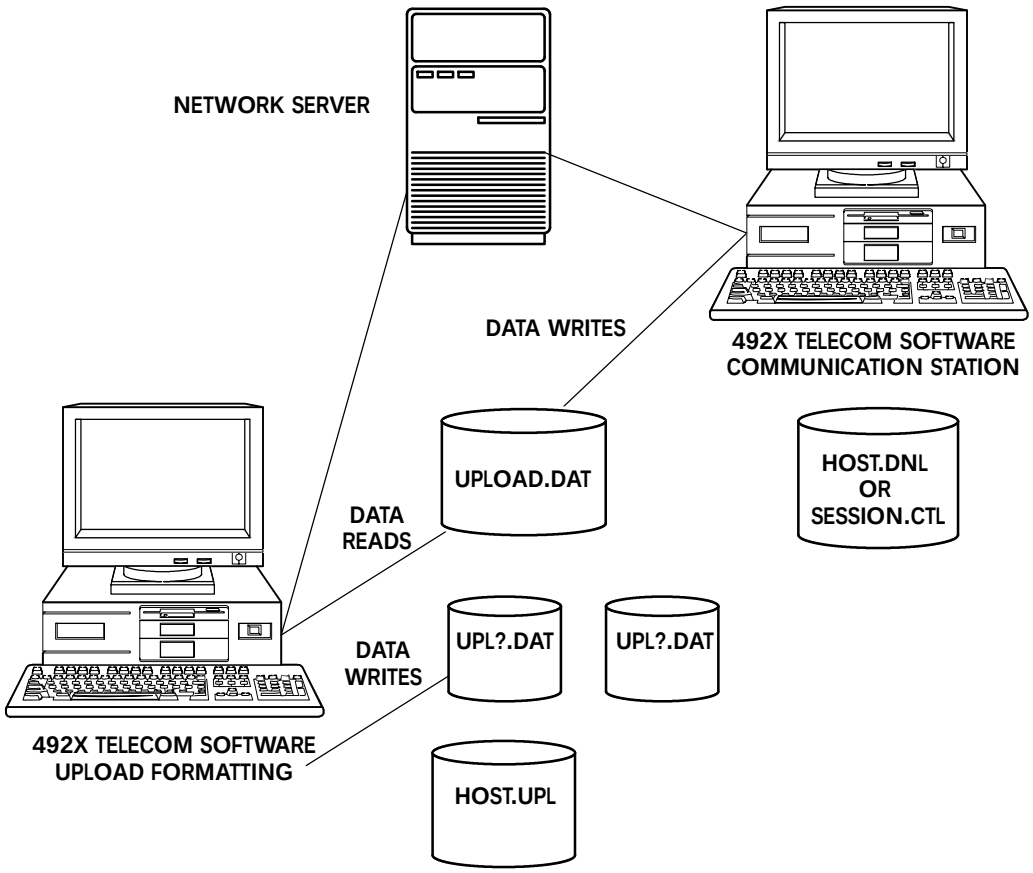

Figure K-3
Example 3

" **NOTE:** *Review 4920 Guide for concurrent upload formatting description and usage.*

- *4920 software installed on local PC*
- *4920 executables and database files (*.idx and *.dat) should reside on local PC hard drive.*
- *Network software installed*
- *Minimum 520K free RAM (See Table K-1)*

- *4920 software download formatters need to point to network drive. 4920 Download directory and Boot directory would point to the network drive. Examples:*

  PLNFMT N:\HHCDATA\INCLUDE.DAT N:\HHCDATA\DOWNLOAD\INCLUDE.INC
  HOSTDNF N:\HHCDATA\HOST.DNL
  SESSFMT N:\HHCDATA\SESSION.CTL

- *4920 Software Upload Formatters can pull the files in from the network or a second PC on the network can be installed with the 4920 and processing of upload concurrently by accessing the "raw" upload file on the network drive. Examples:*

  HOSTUPF N:\HHCDATA\UPLOAD.DAT N:\HHCDATA\HOST.UPL
  UPLFMT N:\HHCDATA\UPLOAD.DAT

- **** SYSTEM.CTLs UPLDIR should be set to N:\HHCDATA\UPLOAD\ add to SYSTEM.CTL- UPLOADFNAME=N:\HHCDATA\UPLOAD.DAT***

Memory Requirements

Current 4920 memory requirements depend on the number of routes the system is to support. Table K-1 shows the memory requirements for the current 4920 product versus 4920 Lite:

Table K-1
Memory Requirements

Number of Routes	492X	492X Lite
0 - 100	520K	300K
101 - 200	525K	320K
201 - 300	545K	340K
301 - 400	565K	360K
401 - 500	590K	385K
501 - 600	620K	415K

492X Lite frees up approximately 200K of conventional memory.

492X Lite Functionality and Schedule

The 492x Lite is packaged as only executable, the 492x Telecommunications Software (Full Feature) must be ordered and installed before

the 492x Lite can be used. 492x Lite was developed by taking the current screen handling package out of the core programs used by the 492x System. Currently the only screen shown in the 492x Lite is the LOG.DAT file displayed to the monitor. There is a schedule in place to have a less sophisticated screen added to the 4920 Lite system. Below is the current schedule for 4920 Lite:

	NPN
4920DOS Lite	215-428-001 . . $249
4921DOS Lite	215-430-001 . . $249
4920OS2 Lite	
4921OS2 Lite	

New 4920 Parameters:

4920 has had new parameters added to the system to provide more functionality for operations on a network. The parameters were added to the SYSTEM.CTL file (Miscellaneous). New parameters:

Table K-2
New 4920 Parameters

Parameter	Input	Description
Error Log File Name	<filename>	All 4920 programs will log critical errors to the file named in this field. No filename logging will not occur.
Network Watch Flag	<filename'+/-'>	The presence (+) or absence(-) of this file will cause the NCP4920.EXE program to end communications.
Do *not* halt programs on an Error	Yes/No	Yes - Critical Errors will not wait for User Intervention to proceed. All errors will be logged to the Error Log File Name if parameter has been entered.
UPLOADFNAME*	<filename>	This parameter alters the default name of the raw upload file. 492x default name is UPLOAD.DAT, this name could be altered to point to a network drive, such as UPLOADF-NAME=N:\UPLOAD\UPLOAD1.DAT
UPLOADBACK**	Yes/No	This parameter keeps a backup of the raw upload file. It will use the UPLOAD.DAT filename or UPLOADFNAME and add the extension "BKK."

** UPLOADFNAME: This parameter has to be added to the SYSTEM.CTL using a text editor. This can be added anywhere within the file, preferably at the end, and should read UPLOADFNAME=<drive:path\filename>*

*** UPLOADBACK: This parameter has to be added to the SYSTEM.CTL using a text editor. This can be added anywhere within the file, preferably at the end.*

Additional Parameters

In order to prevent the 492x software halting on some "Fatal Error Conditions," use the Telecommunications RETRIES parameter in the SYSTEM.CTL file. This parameter will attempt to recover from communication errors to the 4920 Metacomp Board or to the 498x Controller in a 4921 system. The current active terminals will be flagged as "BAD," however the 492x will reinitialize the system and continue. This activity is generally seen with the 4921 "170x" Read Errors or the 4920 "1505" Bus Timeout Errors. This type of error condition can be received if the PC processor is being shared by multiple tasks.

Concurrent Upload Formatting

4920 supports two types of upload data formatting: batch processing and concurrent processing.

Batch Processing: Refers to the processing of upload data *after* the 492x communications program has ended (4920 upload formatting program HOSTUPF.EXE).

Concurrent Processing: Refers to the processing of the upload data while communications is currently running (4920 upload formatting program UPLFMT.EXE). This is normally used on an OS/2 operating system, however changes made to the 4920 software since version 1.11 have allowed this process to function on a network operating system using two or more PCs and a server.

This appendix will only refer to the concurrent upload formatting (UPLFMT.EXE).

A number of new parameters have been added to the upload formatting, the parameters can be entered on the command line when beginning the UPLFMT.EXE program. The usage and description of the parameters are shown:

```
uplfmt: usage: uplfmt {infile | @listfile}
[/c{y|n}] [/f{y|n}] [/rnnn] [/l{y|n}] [/b{y|n}]
[/s{y|n}] [/d{y|n}] [/t{y|n}] [/pprefix] [/xex-
clude prefix] [/n{y|n}] [/h{y|n}] [/zstop flag]
```

Table K-3
Concurrent Upload Format Parameters

Parameter	Input	Description
infile|@listfile	<filename>	This is the *raw* data file, such as UPLOAD.DAT. It can also be a list file which would contain a number of raw data files if the UPLFMT.EXE program is formatting data for multiple 492x Telecommunications programs.
/c	y|n	Process upload concurrently. If a list file is used, the upload formatter will move from upload file to upload file during idle time searching for upload data to process.
/f	y|n	Fixed length records.
/r	nnn	Record length.
/l	y|n	Logical record format.
/b	y|n	Include Bad uploads.
/s	y|n	Include uploads from bad sessions.
/d	y|n	Include Duplicate uploads.
/t	y|n	Delimit records.

Parameter	Input	Description
/p	prefix	Select prefix — Route Number Prefix to Format.
/x	exclude prefix	Exclude prefix — Route Numbers *not* to format.
/n	y|n	Add CR/LF to records. (Not necessary when doing Logical Record Formatting)
/h	y|n	No HALT on errors. (Same as SYSTEM.CTL parameter)
/z	filename	Stop flag file — presence of this file will cause the UPLFMT program to stop.

Concurrent upload formatting has three main operation areas:

1. How does it format?

   The Upload Formatter parses the data one record at a time based on the PL/N header that precedes the data file or the record information contained in the UPLFMT.CTL file. Transaction files have the value "00000" in the record count field and use a reserved PIC Field Description of "X001." Other special PL/N files are: SCNTRL, DEX, and DOS file types.

2. How does it handle Concurrent Upload Formatting?

   There are two main indicators used during concurrent upload formatting which signify to the upload formatter the status of Telecommunications. The first is the "E" record at the end of the UPLOAD.DAT file. The presence of the "E" record signifies to the formatter that Telecommunications has completed and that formatting can end. The second is the absence of the "E" record. In this case the formatter looks at the last "B" record. A "B" record which contains nothing but spaces indicates that upload is still in progress. When the upload phase of telecommunications is complete, the "B" record is rewritten to contain the route ID, time, date, and status information.

   When the formatter determines that it is either at the end of the upload file or that it is waiting for an upload session to complete, the UPLFMT program will display a message "waiting" for a count of 5, and then display "searching." This will continue until the status changes.

3. How does it handle multiple upload files?

To process multiple upload files with the UPLFMT program, a *List File (@)* containing the names of different upload files must be passed on the command line. The UPLFMT formatter checks for the existence of each file and builds a structure for up to 8 different upload files. The formatter will begin searching the first available upload file for data to process. When a waiting condition occurs (as described in part 2) the formatter will attempt to switch to a different upload file in the list. When all files have been initialized the formatter will continually process each of the upload files until either Telecommunications ends (part 2) or the upload formatter is stopped using the stop flag file.

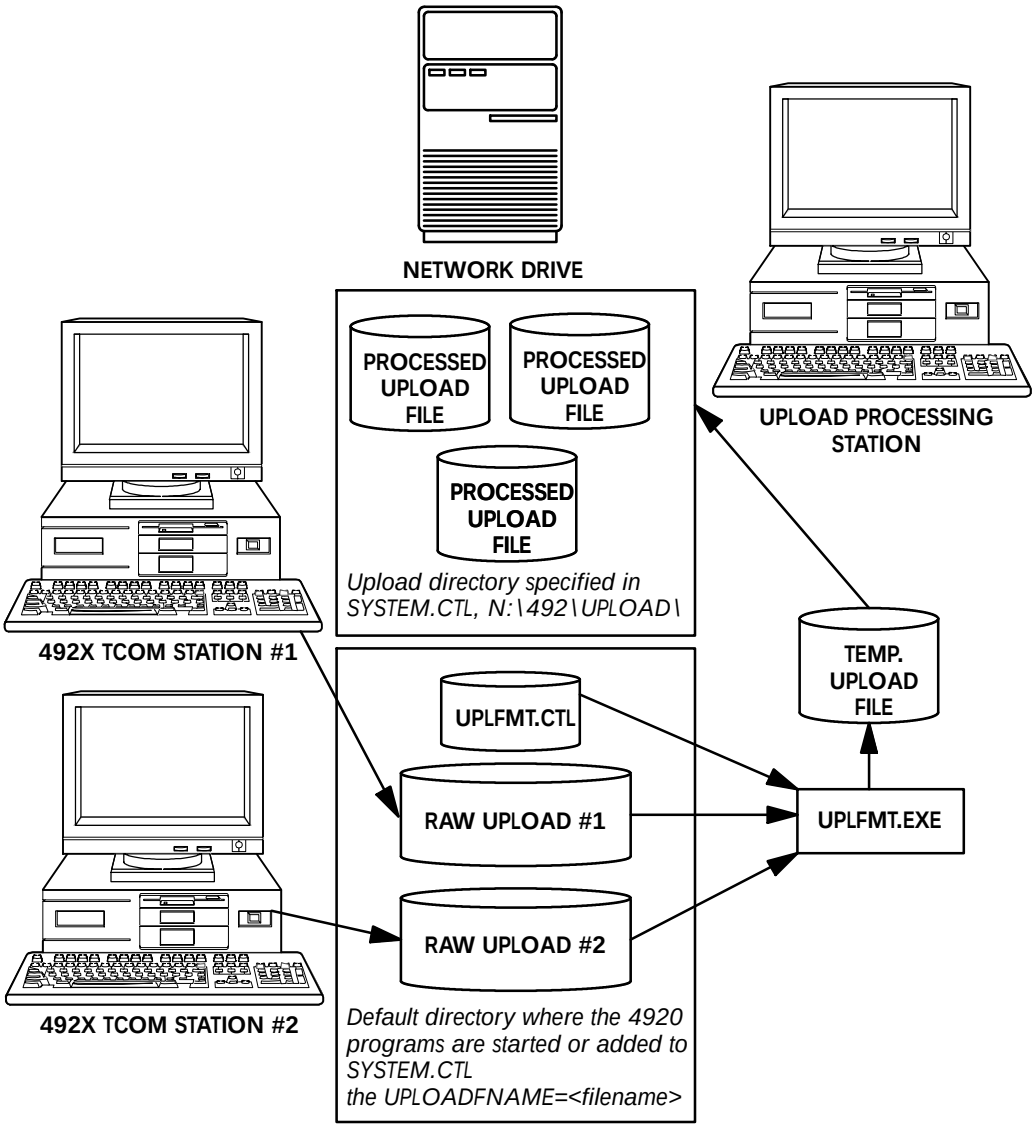

Figure K-4
UPLFMT Program Overview

Memory Managers

When installing the 4920 in an operating system which has device drivers for host emulation or network drive support, it is critical to install an Expanded Memory Manager. The memory manager will assist in two key areas:

1. Freeing up base memory for 4920 operation (loading TSRs in High Memory
2. Excluding memory addresses in which the 4920 Metacomp Card and host or network card access

The key to the memory manager is the exclusions of memory addresses upon PC reboot. The exclusion switch allows memory segments to be blocked from any programs loading into the specific memory location. This ensures that the 4920 and network cards memory locations are unused when the cards are used.

EXAMPLE: This type of usage would be:

Situation: Ethernet card installed at location D000, and 4920 card installed at E000 (4920s MPLD.CTL file would show E0000).

Memory Exclusion of memory regions in CONFIG.SYS file.

Manager: X=D000-D300 X=E000-EFFF.

Explanation: The ethernet card uses approximately 16K memory, the 4920 requires the full 64K memory segment.

Windows 3.x

4920 and 4921 will run under Windows 3.x. There are two main issues:

1. **Available conventional memory:** The 4920 will need conventional memory in order to function. The 4920 has the same memory requirements under windows as in a DOS environment (see "*Memory Requirements*" page K-9). The "Lite" version of 492x may be more functional for customers in a Windows environment.
2. **Create a Windows PIF file:** A Program Interface File will allow the 4920 or 4921 programs to run within the Windows environment. To create a PIF, select the Main Program Group, and double click on the PIF Editor. For each 4920 program create a PIF file. The actual 4920 communication programs are the most critical programs which need a PIF file.

The major parameters in the PIF file are:

- *Display Usage:* Select "Windows" to bring the 4920 programs in a "windows" view.
- *Execution:* Background Execution should be checked.
- *Close Window on Exit:* This should be checked.
- *multitasking Options:* Background and Foreground Priority should be above 500.
- *Memory Options:* Lock Application Memory should be checked, for better performance.

OS/2 Installation

The 4920 will run in an OS/2 environment with a PS/2 machine using the 4920OS2P software. The 4920 OS/2 software and Micro Channel card are the main components. The 4920 OS/2 software comes with a device driver that which communicates between the 4920 software and the Micro Channel board.

- Micro Channel Card Installation

The 4920 Micro Channel card is shipped with factory default settings:

```
Port A - NPCP/LAN Protocol, RS -485
Port B, C, D - ADCCP Modem Communications, TTY Modem
  or Direct Connect, and Ymodem Modem or Direct Con-
  nect.
```

Any changes to the port configurations can be found in Section 3 in Volume A of the *Models 4920 and 4921 Series of PC Telecommunication Packages User's Guide NPN: 961–021–011*.

- MPLD.SYS Device Driver

The 4920 Device Driver is loaded during OS/2 Start-up, the driver MPLD.SYS must be added to your CONFIG.SYS. The MPLD.SYS driver has three parameters associated with its installation:

```
MPLD.SYS <interrupt><Expanded Memory><Extended Memory>
```

Interrupt 1-15 (PS/2 Specific: 10, 11, 12, 14, and 15 are usually open)

Extended Memory: This value is specific to how many 4920 Micro Channel boards and up to 14 Meg of RAM are installed on the PC. Valid values are 1-15. Table K-4 will assist in selecting a proper value.

Table K-4
Valid Extended Memory Parameters

PS/2 RAM	1-4920 Card	2-4920 Cards
4 Meg	6 through 15	6 through 12
6 Meg	8 through 15	8 through 12
8 Meg	10 through 15	10 through 12
12 Meg	14 or 15	Currently not supported
16 Meg	Currently not supported	Currently not supported

Expanded Memory: Valid Values are C000, C400, C800, CC00, D000.

MPLD.SYS when installed will interrogate the system to automatically configure itself for the number of 4920 Micro Channel boards installed.

Multiple 4920 Micro Channel Boards

The 4920 running under the OS/2 operating system on a Micro Channel hardware system can support multiple communication programs and control files. The following items outline the areas which need to be addressed using a multiboard system.

MPLD.SYS: The device driver will automatically configure itself for a multiboard installation.

Directory Structure: The 4920 system should be installed in separate directories. However, the paths in the SYSTEM.CTL file for DOWNLOAD, UPLOAD, and BOOT can be shared between both systems by setting up a common directory.

Download Build: The download database build using the SESSFMT.EXE or HOSTDNF.EXE can be done within one directory, and the database files (*.idx and *.dat) can be copied to the second 4920 system (if a common download directory is shared between the systems.)

If you are using autodial or answer records in their download build, you must be aware of the configuration of the ports for each 4920 sys-

tem. It may be more practical that each 4920 system use separate autodial or answer records. However, if the same autodial or answer records are being used, *both 4920 card jumpers and COMM.CTL files must match exactly.*

If you are using both systems as autoanswer only, a "rollover" phone system must not be used between TTY and ADCCP ports. Two separate "rollover" lines would have to be used.

Upload Processing: The UPLFMT.EXE and HOSTUPF.EXE programs can be used with a "list" file to process uploads from both systems into a common upload directory.

NCP4920: The communications program NCP4920.EXE reads in the COMM.CTL file (use EDITelecommunicationsM.EXE to edit the COMM.CTL file) to configure the software for which 4920 Micro Channel card to attach. The device or card which the NCP4920.EXE program is connected to is defined in the *COMM.CTL Host Port*. The *default* Host Port is:

BUS,mpld1: This refers to the 4920 Micro Channel card #1. To configure a 4920 to communicate using card #2, the *mpld1* in COMM.CTL *must* be changed to *mpld2*.

Efficiency: Running two 4920 cards does not slow down the overall communication throughput, actually more data is transferred in the same amount of time. However, looking at a single port on a card you can expect the throughput to decrease by approximately 15%.

Table K-5
Communication Throughput

	4920 #1		4920 #2	TOTAL
	LAN	**ADCCP**	**LAN**	**DATA**
1 Board 8 Meg RAM	2900 cps	800 cps		3700 cps
1 Board 12 Meg RAM	3050 cps	825 cps		3875 cps
2 Boards 8 Meg RAM	2800 cps	750 cps	2700 cps	6250 cps

DOS File Handling

492x Upload Formatting

Customers using NORAND HHCs have expressed the need to upload and download DOS style data files (binary or ASCII) up to the PC through the NORAND Telecommunications handlers. An example of a binary file is a D-Base file or a Lotus 123 file. Since our terminals can be used as portable PCs running DOS, there is no limit to the types of files that could be sent in an upload.

The upload formatter, previous to 4920 and 4921 Version 1.15 and 4922 Version 1.01 could handle TRANSACTION files, DEX files, and files with fixed length records but it could not process binary or ASCII data in a DOS type file structure. (File characteristics, data, and possibly name are transferred from the HHC to the PC.)

This problem has been resolved with an enhancement to the 492x upload formatter.

To allow the upload formatters to recognize this new type PL/N file the following lines must be added to the UPLFMT.CTL file:

```
FILE=DOSFIL
RECTYPE=DOS
```

This DOS file handling is available starting in Version 1.15 of the 4920 and 4921 and Version 1.01 of the 4922.

HHC Changes Needed

Customized changes would have to be made to the HHC to upload or download what we refer to as a DOS file type. A DOS file uses a predefined PL/N header and an information record that is *prefixed* onto the beginning of the actual file being uploaded. The header and information record is defined as follows:

```
<DDOSFIL00001X0??>FsomefileSnnnnnnn........binary file .......~~
```

<*DDOSFIL:* The *PL/N filename* is arbitrary as long as it matches the *file* defined in the UPLFMT.CTL file. Only one “DOS FILE” name is allowed in the 492x systems UPLFMT.CTL file at this time.

00001: The record count is always 1 for the information record.

X0??>: The PIC field is always an "X" type and it defines the number of bytes in the *information* record. The information record contains the binary file disk name and the size of the binary file.

Fsomefile: "F" is for the disk file name.
"somefile" is the actual file that the formatter will open to receive the data. It must use standard DOS naming conventions and it can include the path information. An "A" can be used in place of the "F" if it is desired to *append* onto an existing file.

" **NOTE:** *A space separates the information fields.*

Snnnnnnn: "S" is for the *size* of the binary file.
"nnnnnnn" represents the number of bytes in the file *excluding* the "~~".

.....binary file.....: Attached file starts here.

To allow the upload formatters to recognize this new type PL/N file the following lines must be added to the UPLFMT.CTL file:

```
FILE=DOSFIL
RECTYPE=DOS
```

This DOS file handling is available starting in Version 1.15 of the 4920 and 4921 and in Version 1.01 of the 4922.

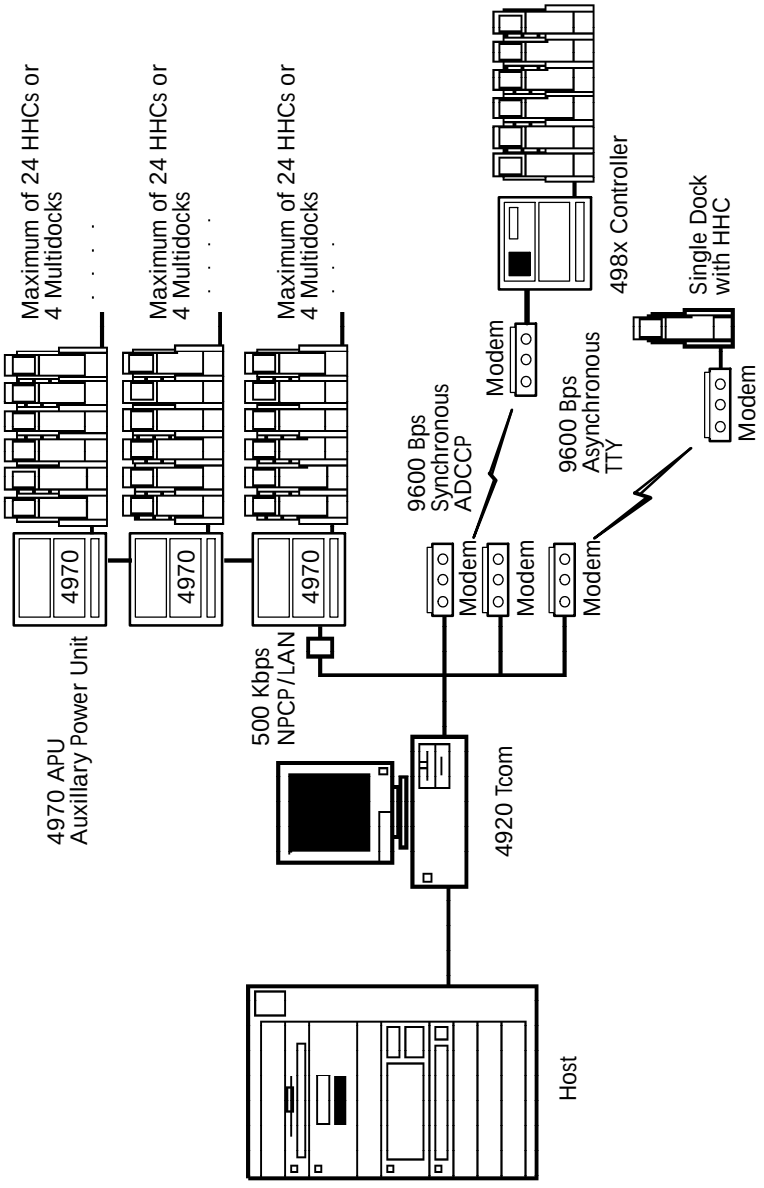

Figure K-5
4920 or 4921 Telecommunications

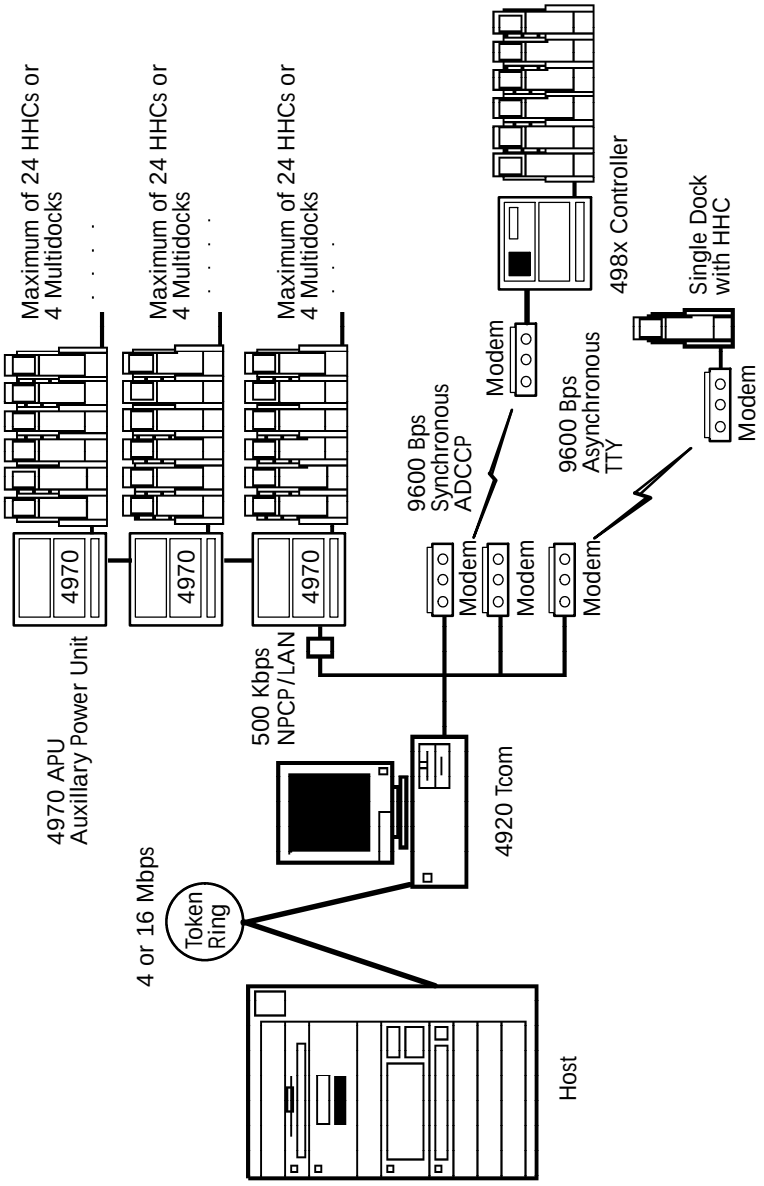

Figure K-6
4920 or 4921 Telecommunications with Token Ring

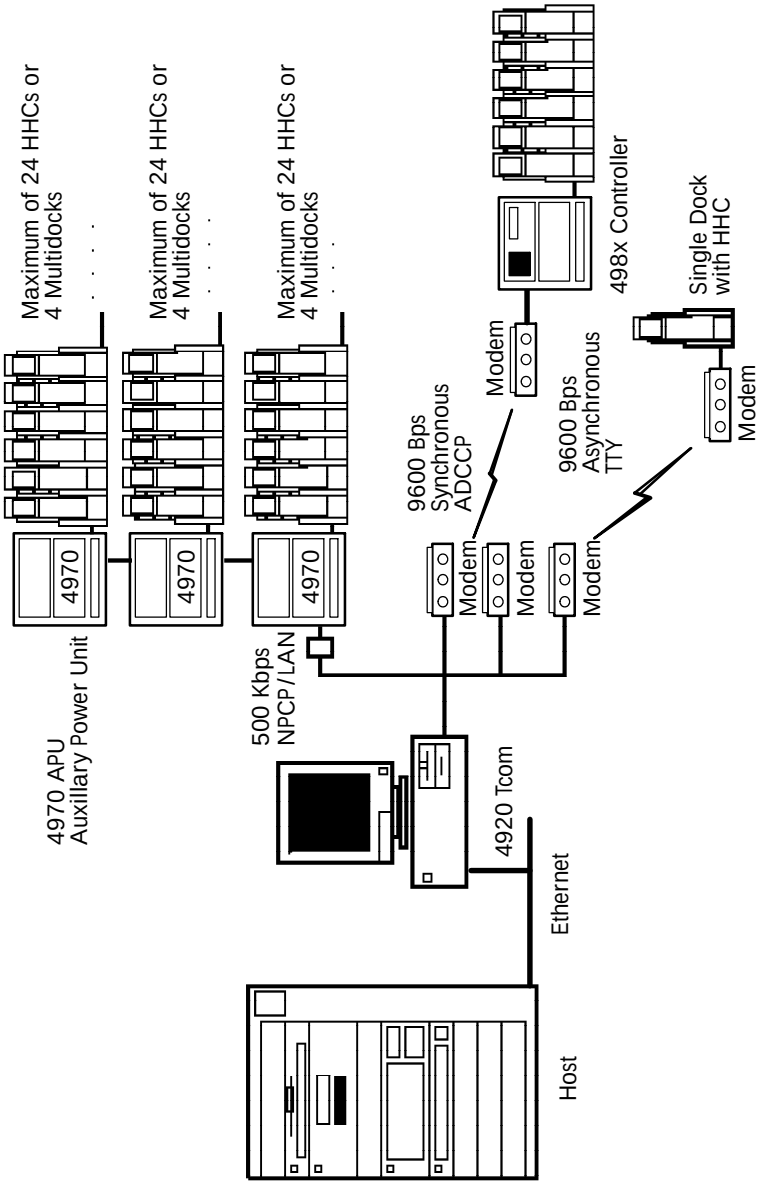

Figure K-7
4920 or 4921 Telecommunications with Ethernet

INDEX

This index covers all topics. Sections 1 through 7 are in the User's Guide, Volume A. Sections 8 through 15 are in the User's Guide, Volume B. Letters refer to the Reference Guide.

Numbers in italics are figures.

NUMBERS

A

B