Host-Micro File Transfer Capability
External Reference Specification

J. L. Larson
December 27, 1984

CHANGE/RESOLUTION FOR THE DOCUMENT WITH DCS LOG ID ARH6298

DOCUMENT TITLE: Host-Micro File Transfer Capability

DOCUMENT TYPE ERS FEATURE _____ PHASE _____

CHANGE/RESOLUTION SUBMITTED BY J. L. Larson DATE 12/27/84

SECTION Technology Design EXTENSION 2708 MAIL STATION ARH244

CHANGE THE DOCUMENT IS

IF CHANGED, APPROVALS AS ON THE ORIGINAL DOCUMENT ARE REQUIRED

- [ ] WITHDRAWN
- [ ] UNCHANGED
- [ ] CHANGED (NO. PAGES _____)
- [XX] ALL

	PRE-DISTRIBUTION APPROVALS	
INTERNAL REVIEWER		
PROJECT MANAGER		
UNIT MANAGER		
SECTION MANAGER J.L. Harrison	JLH	12/27/84

RESOLUTION THE FOLLOWING COMMENTS ARE RESOLVED:

COMMENTS DISTRIBUTION NUMBER	COMMENTS DISTRIBUTION DATE	CONCURRENCE BY THE SUBMITTER OF COMMENTS (IF POSSIBLE)

TO BE FILLED OUT BY DESIGN TEAM DT JLH DESIGN TEAM REFEREE J.L. Harrison

APPROVED FOR DISTRIBUTION BY JLH Harrison DATE 12/27/84

NEW DEFAULT APPROVAL CYCLE 0 WORKING DAYS REDESIGN ___ REIMPLEMENT _____

REMARKS:

This ERS has been updated to reflect NOS 2.4.1 and NOS/VE 1.1.2 systems.

INSTRUCTIONS

CONTROL DATA CORPORATION | DOCUMENT CONTROL FORM

Special Distribution:

R. C. Burkhardt	RVL008
P. L. Derby	SVL110
S. W. Fewer	ARH263
D. B. Folsom	ARH257
J. L. Harrison	ARH244
R. A. Japs	ARH244
A. J. Komor	RSM02S
S. L. Ksander	ARH244
B. G. Palmer	ARH207
W. R. Palmer	ARH213
J. R. Ruble	ARH257
J. R. Spensley	RVL015
R. E. Tate	ARH207
D. D. Thompson	ARH293
D. L. Wondra	ARH247

UMENT
'E ERS

LOG ID ARH6298

LE
ost-Micro File Transfer Capability

STRACT
scribes host package for NOS and NOS/VE to
port micro-mainframe protocol file transfers.

DUCT AFFECTED
NOS, NOS/VE

THOR
J. L. Larson

IL STATION ARH244

o EXTENSION 3061

JECT

TION
Technology Design

or #	o SIP Number(s)
#	
#	o Redesign+ o Reimplementation

	Name	Approval	Date
ernal Reviewer+			
ject Leader	J. L. Larson	JLL	4/24/84
t Manager			
tion Manager	J.L. Harrison	JLH	4/24/84
ign Team eree	J.L. Harrison	JLH	4/24/84

IN

DEFAULT CYCLE 15 WORKING DAYS

o Special Distribution: See attached sheet.

- DCS Mailing List Pasted Here -
(on top of instructions)

+For ARHOPS Use Only!

Referee's Distribution Codes: NHP-B, NSSU-B, NSIT-B

NUMBER	DATE	TYPE	SUBMITTED BY	DEFAULT APPROVAL DATE	CURRENT STATUS: REVIEW	HOLD	APPROVED	WITHDRAWN
1	4/24/84	ORIG	J. L. Larson	5-21-84	R.			
2	5-1-84	NON-CON	W. Palmer	5-21-84				
3	5-22-84	HOLD	DCS	5-21-84		H		
4	5-25-84	comm chg.	J. Larson	NONE			A	
5	12-27-84	comm chg.	J. L. Larson	NONE			A	

Please submit your comments to DCS address SVL160 or ARH230 (as appropriate) before the default approval date.
2441C

ARH6298

. I	SVL069	IOS PROJECT	
. R	SVL100	K.WESTGARD	
. I	SVL102	P.K.KAGAN	
. R	SVL107	P.L.MCNAIR	
. R	SVL207	Y.J.PULAK	
.*1*	SVL107	P.A.SHIRREFF	
. R	SVL143	A.O.HIEBERT	
. I	SVL144	M.A.NOVOTNY	
. I	SVL154	S.T.CHAMPION	
. I	SVL160	R.J.PUMMERS	
. I	SVL199	K.A.DUNN	
. I	SVL192	D.E.LIGHT	
. I	ARH202	L.A.COMPTON	
. R	ARH207	R.E.TATE	
. I	ARH208	(J.C.KNIESCH)	
. R	ARH219	K.L.GOWLAND	
. R	ARH234	R.C.COLLETTE	
. R	ARH242	L.K.JACOBS	
. I	ARH242	D.R.PANSCH	
. I	ARH242	A.E.MADDEN	
. I	ARH242	K.L.SUNDBERG	
. R	ARH242	D.O.HAMNES	
. R	ARH244	R.K.FOSTER	
. R	ARH244	R.A.JAPS	
. R	ARH246	N.E.SUPPORT	
. I	ARH248	R.J.PRIEVE	
. R	ARH248	J.A.WEISS	
. R	ARH263	P.D.PARNELL	
. I	ARH266	L.T.SNAPKO	
. I	ARH267	G.D.PEARMAN	
. R	HQW02X	C.K.STEHMAN	
. I	RPM09T	J.G.CHETTY	
. I	RSM02S	L.A.HOELTJE	
.	RVL005	S.I. miller	
	RVL016	E.E. Ohare	

ARH6298 Special Distribution:

P. L. Derby	SVL110
S W Fewer	ARH263
D. B Folsom	ARH257
J. L Harrison	ARH244
R G Hosmer	RVLO15
R A Japs	ARH244
A. J. Komor	RSMO2S
R. C Lynn	ARH245
P. L. McNair	SVL107
B G Palmer	ARH207
W. R Palmer	ARH213
R J Prieve	ARH248
J R Ruble	ARH257
C D. Sides	HQWO5I
J Sutherland	CANCDD
R E Tate	ARH207
D D Thompson	ARH293
J A Weiss	ARH257
D. L Wondra	ARH247

1.0 INTRODUCTION

This ERS addresses the requirement for protocol file transfer capabilities between a Cyber host and a microcomputer. As part of Control Data's micro accomodation strategy, we need to provide a standard host package available under the NOS and NOS/VE operating systems that will support file transfer using an error checking protocol between a Cyber mainframe and a microcomputer.

1.1 Requirements

- The proprietary communication protocol used by the CONNECT, RMF, and micro-IPF Control Data products should continue to be supported. Although these products have limited features and are very restricted in the types of micros and configurations supported, they do have a significant user base and a worthwhile purpose.

- The standard host package to be developed will reduce CPU utilization by approximately 40% for RMF/CONNECT file transfers (compared to previous host product) and will not exceed 30K field length.

- The public domain protocol XMODEM developed by Ward Christensen should additionally be supported on the Cyber mainframe. This protocol is widely used and is supported by numerous micro communication packages already available in the marketplace for almost any type of microcomputer configuration. This would allow microcomputer users to select a communication package that suits their needs besides providing the XMODEM support for file transfers.

- For support work on advanced systems, the Cyber program needs to be able to detect and dynamically run through Network Products, the Remote Diagnostic Facility and eventually through CDCNET products. This allows all paths of the Cyber to be open to outside communication to micros.

2.0 APPLICABLE DOCUMENTS

- Remote Micro Facility (RMF) Specifications, GID ARH5544.

- Memo to G. M. Schumacher from P. L. Derby on 9/28/83 about RMF/CONNECT strategy. The section written by Ward Christensen providing an overview of the XMODEM protocol has been extracted and attached to this ERS in appendix A.

- User Guide for XMODEM Usage on NOS attached in appendices B and C. (Originally written by Wes Palmer).
- User Guide for XMODEM Usage on NOS/VE attached in appendices D and E. (Originally written by Wes Palmer).
- Input Timeout, Typeahead Purging for IAF, DAP ARH6079.
- Input Timeout, Typeahead Purging for VEIAF, DAP ARH6080.
- Multimessage Transparent Mode Support under VEIAF, DAP ARH6081.
- Change Multimessage Transparent Input Timeout, DAP S4657.
- Working Agreement for CYBERNET CONNECT, 8/83.

3.0 FEATURE DESCRIPTION

The features provided by the Protocol File Transfer Facility (PFTF) will include host support of the CONNECT/RMF proprietary protocol and the XMODEM protocol from all communication paths on the Cyber. The PFTF program will be written in Cybil following NOS/VE Cybil coding standards for standardization on both the NOS and NOS/VE operating systems. The UCOPY entry point in PFTF will support the Control Data proprietary protocol used by the CONNECT, RMF, and micro IPF products. The UCOPY command is issued to the Cyber host by the micro portion of these products (micro IPF uses the CONNECT interface to transfer files). The XMODEM entry point in PFTF will support the XMODEM file transfer protocol. The microcomputer user connects to the Cyber as an interactive user and then issues the XMODEM command on the Cyber host to initiate the file transfer. The PFTF program will be organized into overlays to support the different file transfer protocols. This structure will reduce field length requirements and provide a mechanism for easily providing support for additional protocols.

PFTF will be released in binary only on NOS (as part of tailored release), since the CYBIL compiler is not standard on NOS. NOS/VE will be responsible for maintenance of PFTF after this product's development.

3.1 CONNECT/RMF Protocol Support

Variants of the UCOPY program written in FORTRAN currently provide host support on the NOS operating system for the CONNECT, RMF, and micro IPF products. The CONNECT version of UCOPY was originally based on the UCOPY for RMF, but due to separate feature enhancements to CONNECT by CYBERNET and to RMF by the Special Programs Division, UCOPY for CONNECT now differs somewhat from UCOPY for RMF. Currently, the CONNECT product including UCOPY is being upgraded to support both the COMSOURCE NOS V1 system for CYBERNET and Corporate NOS Version 2.1, 2.2, 2.3 systems. UCOPY will be recombined to support both CONNECT and RMF via the same program and binary. The FORTRAN implementation of UCOPY is very difficult to comprehend and maintain. The CONNECT/RMF host protocol support will be converted to a Cybil program (overlay in PFTF) for use on both NOS and NOS/VE operating systems.

The new Cybil version of UCOPY in the PFTF program must be compatible with all versions of RMF and micro IPF diskettes available in SMD. The upgraded version of CONNECT for the CD110 (for CYBERNET V1 and Corporate NOS V2) must work with PFTF on these systems. Previous versions of CONNECT (CD110 V2.0, Apple V1.0, IBM-PC V1.0), which supply the application accounting parameter (micro type) on the UCOPY command and use the RMUGET procedure file, should also work with PFTF on CYBERNET V1 systems. The UCOPY command and parameters for CONNECT and the UCOPY command and parameters for RMF must be supported. The idiosynchrasies of the CONNECT/RMF protocol, network communications, and expectations of the micro CONNECT, micro RMF, and micro IPF programs must continue to be met.

3.1.1 RMF UCOPY Command

The format of the UCOPY command for RMF is as follows.
The addition of keywords on the UCOPY command is new and is not currently used by micro RMF.

```
UCOPY,filenam,direct,type,blksize.
UCOPY,FN=filenam,TD=direct,FT=type,BS=blksize.
```

Parameter	Description
filenam	File name of file to be transferred. Default=LFILE. Filenam may be local or permanent.
direct	Direction of file transfer. R = receive file from host to micro. S = send file to host from micro. Default = S.

Parameter	Description
type	Type of file to transfer. B = binary. T = text (uses current character set). Default = T. RMF support in PFTF will also allow the following file types (not currently used by RMF micro products): A = ASCII (6/12 bit format on NOS) to/from ASCII (8 bit on micro). C = CDC Display Code (on NOS) to/from ASCII (8 bit on micro).
blksize	Logical block size to be used to transfer data, decimal unless B suffix used. (On NOS/VE, B suffix or (8) post radix or any other post radix from 2 to 16 are accepted).

3.1.2 CONNECT UCOPY Commands

The format of the UCOPY command for the upgraded version of CONNECT (for CYBERNET V1 and Corporate NOS V2) is as follows. The addition of Keywords on the UCOPY command is new and is not currently used by micro CONNECT.

```
UCOPY,filenam,direct,type,blksize,micro.
UCOPY,FN=filename,TD=direct,FT=type,BS=blksize,MT=micro.
```

Parameter	Description
filenam	Same as for RMF.
direct	Same as for RMF.
type	Type of file to transfer. A = ASCII (6/12 bit format on NOS) to/from ASCII (8 bit on micro). B = binary. C = CDC Display Code (on NOS) to/from ASCII (8 bit on micro). T = text (uses current character set). Default = T.
blksize	Same as for RMF.

Parameter	Description
micro	Type of micro initiating transfer (used for application accounting by CYBERNET). Also tells UCOPY to use CONNECT rather than RMF specifications. 1 = CD110. 2 = TRS80. 3 = Apple. 4 = IBM PC. 5 = CD114. 6 = Zenith 100.

3.1.3 CONNECT vs. RMF Differences

The additional micro type parameter on the CONNECT UCOPY is required to distinguish between CONNECT and RMF, as the same program and binary is used to support both products. Support of CYBERNET CONNECT versions prior to this is not required. Significant differences in UCOPY support of CONNECT and RMF include the following:

o End of data block and prompt control.

RMF has prompt turned OFF and signals end of data block as the sequence:

```
    +-----+-----+-----+-----+-----+
... ! ETB !CRC1 ! CRC2! CRC3!  CR !
    +-----+-----+-----+-----+-----+
```

where ETB is the Ascii character signalling beginning of the cyclic redundancy check value, and CRC1, CRC2, CRC3 is the cyclic redundancy check value of the block (refer to RMF Protocol Specification, ARH5544).

CONNECT has prompt turned on and signals end of data block as the sequence:

```
    +-----+-----+-----+-----+-----+-----+-----+-----+-----+
... ! ETB !CRC1 !CRC2 !CRC3 ! DC1 !  CR !  LF !  ?  !  b  !
    +-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

where the DC1 Ascii character and following characters signal end of block to CONNECT.

- End of transmission character.

  RMF uses Ascii EOT (04 Hex).

  CONNECT uses Ascii Bell (07 Hex). This does not interfer with control characters interpretted by Data Services Networks (DSN) and international X.25 networks.

- Maximum error retries.

  RMF retries block transfer 3 times before aborting due to error.

  CONNECT retries block transfer 25 times before aborting due to error.

3.1.4 Deviations for RMF/CONNECT Support

The following changes will be made to UCOPY in the PFTF utility, which deviates from the previous FORTRAN implementation of UCOPY. These changes will not affect compatibility with CONNECT, RMF, or micro IPF diskettes.

- The UCOPY command for RMF will also allow file types A and C (refer to section 3.1.1).

- The logical block size parameter on UCOPY previously defaulted to 2000D. This will be changed to default to 2000D for networks, 148D for RDF (non-network), and 628D for CYBERNET TELEX systems.

- The physical block size for network systems was previously set to 140D for RMF I, 200D for RMF II and 628 for CONNECT. This will be changed to 500D for NOS 2.1 and succeeding network systems and 160D for pre-NOS 2.1 network systems. The physical block size for non-network systems was 150D for RMF and will be changed to 148D (CONNECT physical block size of 628 is not changed).

- For non-network systems, if the logical block size specified on the UCOPY command exceeds the physical block size, a warning message will be issued to the user's dayfile. Previously, this condition was not diagnosed. Data loss could occur with no indication of a possible cause.

- UCOPY will access permanent files when sending from CYBER to micro. Previously, the file was required to be local.

3.1.5 RMF/CONNECT User Notes

- When using CONNECT or RMF on NOS/VE, if special file attributes need to be set, a SET_FILE_ATTRIBUTES command should be issued before the transfer. For example, when receiving an object library from the micro, issue 'SETFA filename FILE_STRUCTURE=LIBRARY' before the transfer.

- On NOS/VE, a text file should not be transferred using the RMF II NOS to NOS option (intended for binary files). When this file is transferred back to NOS/VE, RECORD_TYPE=UNDEFINED is set and it is very difficult to get the file back to RECORD_TYPE=VARIABLE.

- When using RMF II UCOPY via TELENET, the block size should be set to a maximum of 256.

3.2 XMODEM Protocol Support

An overlay in the PFTF program will support the XMODEM file transfer protocol. The XMODEM entry point in PFTF will be executed when the microcomputer user interactively issues the XMODEM command on the Cyber host to initiate the file transfer.

A brief description of the XMODEM file transfer protocol is attached in appendix A. To correctly support end cases of this protocol, deficiencies must be corrected in both the NOS and NOS/VE operating systems and in the Network Products. DAPs ARH6079, ARH6080, ARH6081, and S4657 address these deficiencies with transparent input mode and with input timeout and typeahead purging. Also several NOS/VE page width and attribute problems described in PSRs NV0F241, NV0F242, and NV0F243 must be corrected for XMODEM protocol support and for CONNECT/RMF/micro IPF protocol support. Implementation of solutions to these deficiencies and problems is required to support these protocols. The initial release of PFTF will workaround the lack of timed input in IAF and NOS/VE. When timed input is available, these workarounds can be removed. The NOS 2.4.1 and NOS/VE 1.1.2 release will require the installation of CCP code from SOLVER (DAP S4567) if that code is not integrated into the NOS 2.4.1 release.

XMODEM will not initially work on non-Network (RDF) ports because of a parity problem. When this problem is fixed, XMODEM should function through RDF.

A user guide for XMODEM usage is attached in Appendices B and C (NOS) and Appendices D and E (NOS/VE). XMODEM can be used with some inconvenience on pre-NOS 2.4.1 systems (ie: those without code for DAP S4567). XMODEM does not support pre-NOS 2.1 Networks at all.

Since variations of the Christensen protocol have been implemented in microcomputer products, the XMODEM protocol in the PFTF utility will be defined by the microcomputer communication packages that correctly work with it. The following microcomputer communication packages will be verified for use with XMODEM in the PFTF utility:

- o ASCII Express PRO on Apple II, II+, or IIe
- o ASCII PRO on IBM PC
- o PC-TALK III on IBM PC
- o CROSSTALK 3.5 on IBM PC (versions prior to 3.5 will not work).
- o ZLYNK on Z100
- o ZSTEM on Z100
- o TELED110 on CD110
- o MicroCom on CD110
- o MacTerminal for Apple MacIntosh

3.2.1 XMODEM Command

The positional and keyword formats of the XMODEM command are as follows:

```
XMODEM,filename,direct,type,lf,sp.
XMODEM,FN=filenam,TD=direct,FT=type,LF=lf,SP=sp.
```

Keyword	Parameter	Description
FN	filenam	File name of file to be transferred.
TD	direct	Direction of file transfer. (1) S = send file from host to micro. R = host to receive file from micro.

(1) Note that TD=S/R parameter value is opposite direction for XMODEM as for UCOPY. Since the user must initiate the XMODEM command, it is felt that this definition is clearer (user directs XMODEM on host to send or recieve). This will not interfer or be incompatible with UCOPY usage, since micro user does not initiate or see the UCOPY command.

Keyword	Parameter	Description
FT	type	Type of file to transfer. A = ASCII text (6/12 bit format on NOS) to/from ASCII text (8 bit on micro). B = Cyber binary. T = Text (uses current character set). C = CDC Display Code (on NOS) to/from ASCII text (8 bit on micro). E = ASCII text (8/12 bit format on NOS) to/from ASCII text (8 bit on micro). M = Micro binary. O = Object library (NOS/VE only). S = Automatic Selection (receive only). Special control characters in the transmitted block determine the file type. If control characters are present but unrecognized, file type is assumed to be M. If control characters are not present, file type is assumed to be T. For NOS/VE, options A, C, E are equivalent to option T (text).
LF	lf	For text files sent from host to micro (TD=S), this parameter may optionally be specified on the XMODEM command to select end-of-line as carriage return or carriage return and line feed. (Parameter ignored if TD=R selected). YES = line feed is needed (CR LF). NO = line feed is not needed (CR). Default = YES.

Keyword	Parameter	Description
		For NOS, a parameter value beginning with Y or N will be accepted. For NOS/VE, this parameter is a boolean value (YES/NO, TRUE/FALSE, ON/OFF) or variable. This parameter will not be prompted for.
SP	sp	This parameter is provided for micros which cannot handle the special file type block. Default = NO. NO = Do not suppress special file type block. YES = Suppress special file type block. This parameter is only valid on the XMODEM command. It will not be prompted for.

If the parameters FN, TD, and FT are not specified on the XMODEM command, the user will be prompted individually for those not specified (no defaults are assumed). The following prompts will be sent for the corresponding absent parameters:

Keyword	Parameter	Prompt
FN	filenam	Please enter the file name.
TM	direct	Host to Send (S) or Receive (R) a file?
FT	type	NOS dialog for FT parameter: Is the file - B = Cyber binary M = Micro binary T = Text file ? If TD=R has been selected, the following option is also available in the first prompt:

Keyword	Parameter	Prompt
		S = Auto select If the response T (text) is selected, a second prompt requests character set information: Is the text on the Cyber - A = 6/12 Ascii (upper/lower case) C = CDC Display code (upper case only) E = 8/12 Ascii character set ? NOS/VE dialog for FT parameter When TD = R: Is the file - T = Text file B = Cyber binary O = Object library M = Micro binary S = Auto Select On NOS/VE when TD=S, no dialog is prompted, since file type is determined by the file attributes.

3.3 PFTF Error Messages

Note: For NOS/VE, system messages generated for command and parameter cracking are not included here, as these messages are documented in the appropriate system manual.

Message	Applies To	Significance
Already existing text file must have record_type=variable - filenam	VE - UCOPY/XMODEM	When receiving file filenam from a micro, the record type of the opened mainframe file doesn't match that specified.
Already existing binary file must have record_type= undefined - filenam	VE - UCOPY/XMODEM	When receiving file filenam from a micro, the record type of the opened mainframe file doesn't match that specified.

Message	Applies To	Significance
Already existing binary file must have file_contents =object - filenam	VE - UCOPY/XMODEM	When receiving file filenam from a micro, the file contents of the opened mainframe file doesn't match that specified.
CRC=hhhh.	NOS, VE - XMODEM	Informative message after file transfer giving cyclic redundancy check in hex. (CRC is computed until the EOI character on a binary file is encountered or until the CR after #EOI is encountered on a text file).
Data type is binary, command indicated text.	VE - UCOPY	Data type indicated by the first character in block must match file type specified on command, since file attributes are set when file is opened.
Data type is text, command indicated binary.	VE - UCOPY	Data type indicated by the first character in block must match file type specified on command, since file attributes are set when file is opened.
filenam is direct access.	NOS - UCOPY/ XMODEM	Informative message indicating the file filenam being sent from Cyber to micro was found as a direct access permanent file.
filenam is empty.	NOS, VE - UCOPY/XMODEM	File filenam to be sent from CYBER to micro is empty.

Message	Applies To	Significance
filenam is indirect access.	NOS - UCOPY/ XMODEM	Informative message indicating the file filenam being sent from Cyber to micro was found as an indirect access permanent file.
filenam is local.	NOS - UCOPY/ XMODEM	Informative message indicating the file being sent from Cyber to micro was found as a local file.
filenam is not a text file.	VE - UCOPY	File filenam is not text when text file option selected (record type not variable).
filenam is not readable.	NOS, VE - UCOPY/XMODEM	File filenam does not allow read mode.
filenam is not writable.	NOS, VE - UCOPY/XMODEM	File filenam does not allow write mode.
filenam not found.	NOS - UCOPY/ XMODEM	Informative message indicating the file filenam to be sent from Cyber to micro could not be found.
Incorrect block size specified - blksize.	NOS - UCOPY	Block size is invalid. (.LT. 10. or .GT. 2000.).
Incorrect block type received.	NOS, VE - UCOPY	Block type transferred was illegal.
Incorrect character in binary transfer - xxx.	NOS, VE - UCOPY	SUB or RS Ascii special character detected during binary transfer.
Incorrect control character encountered.	NOS, VE - UCOPY	Unexpected or invalid control character encountered.
Incorrect file name - filenam.	NOS-UCOPY/ NOS, VE - XMODEM	File name filenam is invalid.
Incorrect file type - type.	NOS - UCOPY/XMODEM	File type is invalid.

Message	Applies To	Significance
Incorrect horizontal tab sequence.	NOS, VE - UCOPY	Error in HT sequence.
Incorrect LF parameter specified.	NOS - XMODEM	Parameter value must begin with Y or N to be interpretted as YES or NO.
Incorrect new block size in NAK.	NOS, VE - UCOPY	New block size invalid. (.LT. 10. or .GT. 2000.).
Incorrect sequence number received.	NOS, VE - UCOPY	Sequence number received is not possible (not within expected sequence number + or - 1 range).
Incorrect transfer direction - direct.	NOS - UCOPY/XMODEM	File transfer direction is invalid.
Loader error - initialization overlay.	NOS- UCOPY/XMODEM	Initialization overlay load failed.
Loader error - protocol overlay.	NOS- UCOPY/XMODEM	Protocol overlay load failed.
No initial NAK received.	NOS, VE- XMODEM	Failed to receive initial NAK.
Positional not allowed after keyword.	NOS- UCOPY/XMODEM	Positional parameter cannot follow keyword parameter.
SUB incorrect as last character in block.	NOS, VE - UCOPY	SUB Ascii special character not allowed as last character in block.
Terminal interrupt.	NOS, VE- UCOPY/XMODEM	User interrupted transfer.
Too many transfer errors.	NOS, VE- UCOPY/XMODEM	Excessive communication errors caused transfer failure.
Unexpected end of transmission.	NOS, VE- UCOPY	EOT was received unexpectedly.

Message	Applies To	Significance
Unknown parameter - parm.	NOS-UCOPY/XMODEM	Parameter was unrecognized.
Warning - maximum logical block size exceeds physical block size.	NOS-UCOPY	Warning message that logical block size must be less that or equal to physical block size if non-network system.
XMODEM unavailable on pre-2.1 networks.	NOS-XMODEM	Message issued when XMODEM executed on a network terminal and PFTF was assembled with options indicating pre-NOS 2.1/580 networks on system.

Appendix A:

MODEM PROTOCOL OVERVIEW 178 Lines, 7.5k

1/1/82 by Ward Christensen. I will maintain a master copy of this. Please pass on changes or suggestions via CBBS/Chicago at (312) 545-8086, or by voice at (312) 849-6279.

NOTE this does not include things which I am not familiar with, such as the CRC option implemented by John Mahr.

Last Rev: (none)

At the request of Rick Mallinak on behalf of the guys at Standard Oil with IBM P.C.s, as well as several previous requests, I finally decided to put my modem protocol into writing. It had been previously formally published only in the AMRAD newsletter.

Table of Contents

------ 1. DEFINITIONS.

(soh) 01H
(eot) 04H
(ack) 05H
(nak) 15H
(can) 18H

------ 2. TRANSMISSION MEDIUM LEVEL PROTOCOL

Asynchronous, 8 data bits, no parity, one stop bit.

The protocol imposes no restrictions on the contents of the data being transmitted. No control characters are looked for in the 128-byte data messages. Absolutely any kind of data may be sent - binary, ASCII, etc. The protocol has not formally been adopted to a 7-bit environment for the transmission of ASCII-only (or unpacked-hex) data, although it could be simply by having both ends agree to AND the protocol-dependent data with 7F hex before validating it. I specifically am referring to the checksum, and the block numbers and their ones-complement.

Those wishing to maintain compatibility of the CP/M file structure, i.e. to allow modemming ASCII files to or from CP/M systems should follow this data format:

* ASCII tabs used (09H); tabs set every 8.
* Lines terminated by CR/LF (0DH 0AH)
* End-of-file indicated by ^Z, 1AH. (one or more)
* Data is variable length, i.e. should be considered a continuous stream of data bytes, broken into 128-byte chunks purely for the purpose of transmission.
* A CP/M "peculiarity": If the data ends exactly on a 128-byte boundary, i.e. CR in 127, and LF in 128, a subsequent sector containing the ^Z EOF character(s) is optional, but is preferred. Some utilities or user programs still do not handle EOF without ^Zs.
* The Last block sent is no different from others, i.e. there is no "short block".

----- 3. MESSAGE BLOCK LEVEL PROTOCOL

Each block of the transfer looks like:
(SOH)(blk #)(255-blk #)(--128 data bytes--)(cksum)
in which:

(SOH) = 01 hex
(blk #) = binary number, starts at 01 increments by 1, and wraps OFFH to 00H (not to 01)
(255-blk #) = blk # after going thru 8080 "CMA" instr, i.e. each bit complemented in the 8-bit block number. Formally, this is the "ones complement".
(cksum) = the sum of the data bytes only. Toss any carry.

----- 4. FILE LEVEL PROTOCOL

---- 4A. COMMON TO BOTH SENDER AND RECEIVER:

All errors are retried 10 times. For versions running with an operator (i.e. NOT with XMODEM), a message is typed after 10 errors asking the operator whether to "retry or quit".

Some versions of the protocol use (can), ^ASCII X, to cancel transmission. This was never adopted as a standard, as having a single "abort" character makes the transmission susceptible to false termination due to an (ack) (nak) or (soh) being corrupted into a (can) and canceling transmission.

The protocol may be considered "receiver driven", that is, the sender need not automatically re-transmit, although it does in the current implementations.

---- 4B. RECEIVE PROGRAM CONSIDERATIONS:

The receiver has a 10-second timeout. It sends a (nak) every time it times out. The receiver's first timeout, which sends a (nak), signals the transmitter to start. Optionally, the receiver could send a (nak) immediately, in case the sender was ready. This would save the initial 10 second timeout. However, the receiver MUST continue to timeout every 10 seconds in case the sender wasn't ready.

Once into a receiving a block, the receiver goes into a one-second timeout for each character and the checksum. If thereceiver wishes to (nak) a block for any reason (invalid header, timeout receiving data), it must wait for the line to clear. See "programming tips" for ideas.

Synchronizing: If a valid block number is received, it will be: 1) the expected one, in which case everything is fine; or 2) a repeat of the previously received block. This should be considered OK, and only indicates that the receivers (ack) not glitched, and the sender re-transmitted; 3) any other block number indicates a fatal loss of synchronization, such as the rare case of the sender getting a line-glitch that looked like an (ack). Abort the transmission, sending a (can).

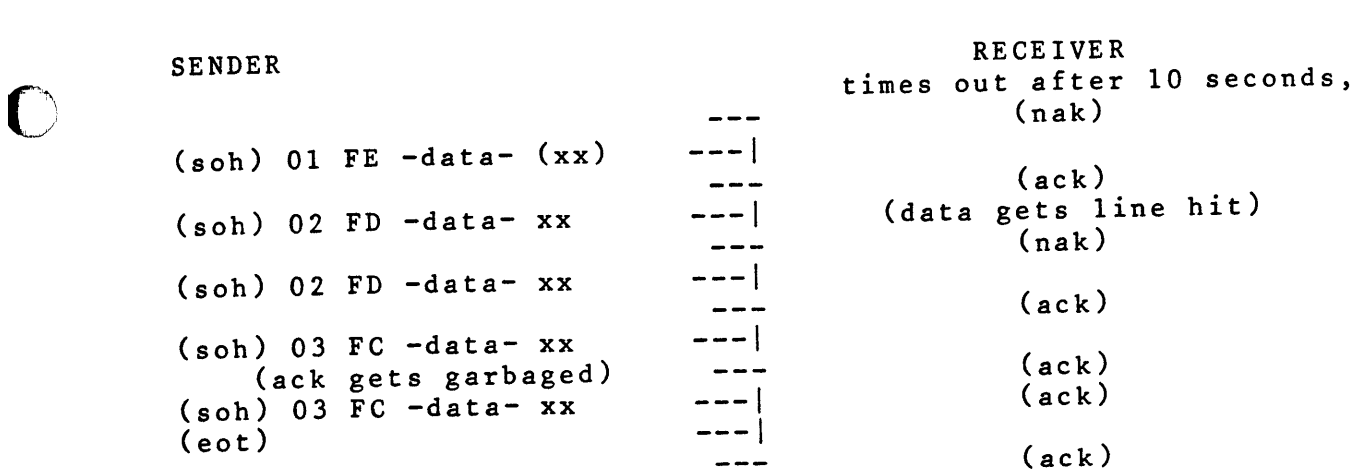
---- 4C. SENDING PROGRAM CONSIDERATIONS.

While waiting for transmission to begin, the sender has only a single very long timeout, say one minute. In the current protocol, the sender has a 10 second timeout before retrying. I suggest NOT doing this, and letting the protocol be completely receiver-driven. This will be compatible with existing programs.

When the sender has no more data, it sends an (eot), and awaits an (ack), resending the (eot) if it doesn't get one. Again, the protocol could be receiver-driven, with the sender only having the high-level 1-minute timeout to abort.

----- 5. DATA FLOW EXAMPLE INCLUDING ERROR RECOVERY

Here is a sample of the data flow, sending a 3-block message. It includes the two most common line hits - a garbaged block, and an (ack) reply getting garbaged. (xx) represents the checksum byte.



----- 6. PROGRAMMING TIPS.

* The character-receive subroutine should be called with a parameter specifying the number of seconds to wait. The receiver should first call it with a time of 10, then (nak) and try again, 10 times.

After receiving the (soh), the receiver should call the character receive subroutine with a 1-second timeout, for the remainder of the message and the (cksum). Since they are sent as a continuous stream, timing out of this implies a serious line glitch that caused, say 127 characters to be seen instead of 128.

* When the receiver wishes to (nak), it should call a "PURGE" subroutine, to wait for the line to clear. Recall the sender tosses any characters in its UART buffer immediately upon completing sending a block, to ensure to glitches were misinterpreted.

The most common technique is for "PURGE" to call the character receive subroutine, specifying a 1-second timeout, and looping back to PURGE until a timeout occurs. The (nak) is then sent, ensuring the other end will see it.

* You may wish to add code recommended by John Mahr to your character receive routine - to set an error flag if the UART shows framing error, or overrun. This will help catch a few more glitches - the most common of which is a hit in the high bits of the byte in two consecutive bytes. The (cksum) comes out OK since counting in 1-byte produces the same result of adding 80H + 80H as with adding 00H + 00H.

Appendix B: XMODEM User Guide for pre-NOS 2.4.1 systems

XMODEM is a program to transfer files between a Cyber and a micro using Christensen protocol. It can be used to transfer text files, Cyber 170 binaries, Cyber 180 binaries, and micro binaries.

Files are transferred as follows:

- Text files have an ASCII CR or CR LF appended to the end of each line (an end of line is a zero byte terminator) when being transferred from Cyber to micro. When transferred from micro to Cyber, carriage returns are translated into zero byte terminators, and ASCII LF's (line feeds) are ignored if they immediately follow a CR. EOR, EOF, and EOI indicators are maintained.

- Cyber 170 and 180 binaries are transferred eight bits at a time. The file is blocked during transmission, with the first character of the block being the length of the block or an EOR/EOF/EOI character. If an EOR/EOF character is first, the next character is again a block length or an EOR/EOF/EOI. The maximum block size is 240 characters. EOR, EOF, and EOI indicators are maintained.

- Micro binaries received by the Cyber are stored as 8/12 ASCII. Each 8 bit ASCII character received is stored on the Cyber as a 12 bit ASCII character with the high order bit set. When being transmitted back to the micro, any zero character is ignored. Thus, NULL's must be represented by 4000B. The high order bit is optional on all other characters. A file processed by FCOPY will meet these requirements (zero byte terminated EOL's will be ignored because they are zero). Using this method, the exact file length can be maintained.

The following are limitations or features that must be considered:

1. As Christensen protocol is an eight bit protocol, parity must be set to NONE. XMODEM automatically issues a TRMDEF,PA=N. However, some data networks, such as NET/ONE may force a specific parity. Some NET/ONE lines are always even parity and cannot be used. Also, the communications line must be capable of passing all 256 ASCII characters.

2. Echoplex must be set to OFF (EP=N). If EP=Y is set, each character the micro sends is echoed back to it. The micro cannot distinguish between echoed data and a response to its last transmission. XMODEM automatically issues a TRMDEF,EP=N. If EP=Y is set, it will have to be manually reset after the transfer, as there is no way to determine the current setting of echoplex.

3. Some micros, such as the APPLE, require the high order bit in each character to be set for certain text files. If a file is transferred as a text file, it will not have the high order bit set. There are two solutions available. First, you could write a simple program on the micro to set the high order bit. Second, if the file is going from micro to Cyber to micro, and is not going to be used on the Cyber, it could be transferred as a micro binary.

4. In transferring text files or Cyber binaries, special characters are used to indicate file markers. For binaries, 375(8) indicates an EOR, 376(8) indicates an EOF, and 373(8) indicates an EOI. For text files, where end of line (EOL) is defined as carriage return (CR) or carriage return and line feed (CR LF), #EOR(EOL) indicates an EOR, #EOF(EOL) indicates an EOF, and #EOI(EOL) indicates an EOI. These characters are always inserted in files being transferred from Cyber to micro. When these characters are encountered as a file is received from the micro, they are translated into the appropriate file markers. To be recognized as file markers in a Cyber binary, the file marker character must occur where a block length is expected. The next character would then be another file marker or a block length. To be recognized in a text file, the file marker characters must occur immediately after end of line.

   These characters have no special meaning if the file is transferred as a micro binary.

5. Currently, due to a deficiency in transparent input, if a file does not have the special EOI character(s), transmisson will hang after the last block is transmitted from micro to Cyber. The last block transmitted from the micro is a single character, EOT. As the Cyber is expecting 132 characters, it cannot respond to the EOT. The micro will repeatedly timeout. When this happens, abort the micro transfer program and enter 131 or more blanks followed by a carriage return. This will fill out the last block. XMODEM will see that the first character of that block is an EOT and shut down. If the file contains the special EOI character (373(8) for binary file or #EOI (CR) for text file), and the file is not being transferred as a micro binary, this problem will not occur. When XMODEM sees the EOI character, it knows to expect only one character in the next block.

   Files transferred to the micro as a Cyber binary or text file always have the special EOI character.

6. The first block transmitted from/to Cyber is of the form:

   SOH+200B / file type / 255D-file type / 123D / zeros

   File type is:
   - 00 - Text
   - 04 - Micro binary
   - 08 - Cyber binary

   This block will be recognized by ASCII Express for the Apple and by XMODEM. If it is not explicitly supported by your micro's transfer program, it will be considered an illegal block and will be NAK'ed. If this block is NAK'ed three times, the transfer will proceed with the data. If the first block received is of this form, and XMODEM is in automatic file type detection mode, the file type in the block will be used to determine the type of transfer. If an illegal file type is found, micro binary is used. If XMODEM is in automatic mode, and no file type block is received, file type text is set.

7. XMODEM does not support pre-NOS 2.1 networks.

8. When receiving a file from the micro with XMODEM, an extraneous error message will sometimes occur after the transfer is complete when using some micros (such as Apple). This is because some micros always send an even number of blocks, so there may be an extraneous block after the EOI character. The message 'Command too long' may occur on NOS. This problem can be fixed when DAP S4657 is implemented.

9. When using XMODEM via TELENET, issue 'set?b0:33,63:0' when in TELENET command mode to set 8-bit ascii mode. This can be done before connecting to the host, or you can escape to TELENET command mode by typing (CR)@(CR). After typing the SET command, you can reenter the host with the CONT command.

10 XMODEM will not work over X.25 networks prior to NOS 2.3 due to problem in X.25 TIP.

Using XMODEM

Log into the Cyber using the appropriate micro communications package (such as ASCII Express on the APPLE). Make sure the line you are using will handle PARITY=NONE. All eight data bits must be passed.

Type:

XMODEM.

You will receive the prompt:

Host to Send (S) or Receive (R) a file?

Enter "S" (no quotes) to send a file from the Cyber to a micro. Enter "R" to receive a file from a micro. You may enter the entire word "SEND" or "RECEIVE".

SEND

If you entered "SEND", the next prompt will be:

Please enter the file name.

The file you specify must be a local file that can be read, and must not be empty. If the file is empty, the prompt for file name will be repeated. If the file you wish to transfer is permanent, you must do a GET or ATTACH before executing XMODEM.

XMODEM will then ask:

Is the file -
T = Text
B = Cyber binary
M = Micro binary

Enter the character that describes the type of file you are sending.

TEXT file

If you indicate the file is a text file, you will be asked:

Is the text on the Cyber -
A = 6/12 Ascii (upper/lower case)
C = CDC Display code (upper case only)
E = 8/12 Ascii character set

Enter the character that describes the character set of the file.

XMODEM will then print out the approximate number of blocks to be transfered.

At this point, you should enter the appropriate commands on the micro (CTRL-G, etc.) to initiate the file transfer. If XMODEM does not receive a NAK after 10 tries, it will abort with the message:

No initial NAK received.

Thus, if you decide to abort the transfer before it has started, enter 11 of any character except NAK (CTRL-U). If you decide to abort the transfer after it has started, enter a CTRL-X (ASCII CAN).

RECEIVE

If you entered "RECEIVE", the first prompt will be:

Please enter the file name.

Enter the file name to be used on the Cyber. It must be a valid NOS local file name, and must have WRITE access.

XMODEM will then ask:

Is the file -
T = Text
B = Cyber binary
M = Micro binary
S = Auto select

If you specified automatic file type detection, the first block received will determine the type of transfer (see note 6 above). If the file type detected is TEXT, 6/12 display code will be used for the character set.

If you indicated the file is a text file (option T), you will be asked:

Is the text on the Cyber -
A = 6/12 Ascii (upper/lower case)
C = CDC Display Code (upper case only)
E = 8/12 Ascii character set

This option determines in which character set the file will be stored.

If option C is selected, characters which cannot be translated to the 64 character set, except for NULL's, will be set to blanks. NULL's will be ignored. Lower case alphabetics will be translated to upper case.

When receiving a file from micro to CYBER, if XMODEM were to adhere to Christensen protocol on pre-NOS 602 systems, it would issue a NAK every 10 seconds until it received a reply. However, on pre-NOS 602 systems or on non-network systems, timed input is not available under IAF, and XMODEM cannot do this. XMODEM rolls out for 20 seconds, issues a NAK, then goes into input mode waiting for a reply. If input is not received, XMODEM cannot regain control, and so cannot issue another NAK after 10 seconds. So, the micro must be prepared to receive within 20 seconds. If the micro is not ready within that time, the transfer will not take place. If this happens, abort the micro program. Then enter CTRL-U (NAK) 14 times. This will cause XMODEM to terminate with the message

Too many transfer errors.

On NOS 2.2/602 to 2.3 systems, XMODEM will wait 20 seconds, send a NAK, check for input for 30 seconds, issue a NAK, etc. The 30 second wait is required to support automatic file transfer on ASCII Express. At future NOS releases, when several operating system deficiencies and problems are corrected, XMODEM will be able to conform to the protocol specifications.

If you decide to abort the transfer after it has started, you must enter characters such that the first character in a block is an EOT. The easiest way to do this is to hold down CTRL-D (EOT) and REPEAT for somewhere around 15-20 seconds. If you enter 264 EOT's, you can be sure you have at least one block in which the first character is an EOT.

Command Parameters

Instead of using the menu mode described above, you may use command parameters. The available parameters are:

FN - Local File Name
TD - Transfer Direction (Send or Receive)
FT - File Type
  A = ASCII (6/12 on NOS) to/from ASCII (8 bit on micro)
  B = Cyber binary
  T = Text (uses current character set)
  C = CDC Display Code to/from ASCII (8 bit on micro)
  E = ASCII (8/12 on NOS) to/from ASCII (8 bit on micro)
  M = Micro binary
  S = Automatic Selection (receive only). If special control characters are transmitted (see note 6 above) but unrecognized, file type is assumed to be M. If control characters are not present, file type is assumed to be T.
LF - Line Feed.
  YES = End of line requires CR LF.
  NO = End of line indicated by CR (no LF needed)
  Default = YES.
  A parameter value beginning with Y or N will be accepted.
  SP - Suppress special file type block.
  NO = Do not suppress special file type block (default).
  YES = Suppress special file type block.
SP - Suppress special file type block.
  NO = Do not suppress special file type block (default).
  YES = Suppress special file type block.

Prompts will be issued for any illegal or missing parameters, except the LF parameter. Parameters may be specified in positional or keyword mode, but a positional may not follow a keyword parameter.

Examples:

```
XMODEM,FN=TEST,TM=SEND,FT=C,LF=YES.
```

would send the CDC Display Code text file TEST, maintaining file markers, with end of lines transferred as CR LF. An equivalent control statement would be:

```
XMODEM,TEST,S,C.
```

To receive file ABC using automatic file type detection, enter:

```
XMODEM,ABC,R,S.
```

To be prompted for FN, TD, FT and to select no line feeds at end of line, enter:

```
XMODEM,LF=N.
```

Appendix C: XMODEM User Guide for NOS 2.4.1 system

XMODEM is a program to transfer files between a Cyber and a micro using Christensen protocol. It can be used to transfer text files, Cyber 170 binaries, Cyber 180 binaries, and micro binaries.

Files are transferred as follows:

- Text files have an ASCII CR or CR LF appended to the end of each line (an end of line is a zero byte terminator) when being transferred from Cyber to micro. When transferred from micro to Cyber, carriage returns are translated into zero byte terminators, and ASCII LF's (line feeds) are ignored if they immediately follow a CR. EOR, EOF, and EOI indicators are maintained.

- Cyber 170 and 180 binaries are transferred eight bits at a time. The file is blocked during transmission, with the first character of the block being the length of the block or an EOR/EOF/EOI character. If an EOR/EOF character is first, the next character is again a block length or an EOR/EOF/EOI. The maximum block size is 240 characters. EOR, EOF, and EOI indicators are maintained.

- Micro binaries received by the Cyber are stored as 8/12 ASCII. Each 8 bit ASCII character received is stored on the Cyber as a 12 bit ASCII character with the high order bit set. When being transmitted back to the micro, any zero character is ignored. Thus, NULL's must be represented by 4000B. The high order bit is optional on all other characters. A file processed by FCOPY will meet these requirements (zero byte terminated EOL's will be ignored because they are zero). Using this method, the exact file length can be maintained.

The following are limitations or features that must be considered:

1. As Christensen protocol is an eight bit protocol, parity must be set to NONE. XMODEM automatically issues a TRMDEF,PA=N. However, some data networks, such as NET/ONE may force a specific parity. Some NET/ONE lines are always even parity and cannot be used. Also, the communications line must be capable of passing all 256 ASCII characters.

2. Echoplex must be set to OFF (EP=N). If EP=Y is set, each character the micro sends is echoed back to it. The micro cannot distinguish between echoed data and a response to its last transmission. XMODEM automatically issues a TRMDEF,EP=N. If EP=Y is set, it will have to be manually reset after the transfer, as there is no way to determine the current setting of echoplex.

3. Some micros, such as the APPLE, require the high order bit in each character to be set for certain text files. If a file is transferred as a text file, it will not have the high order bit set. There are two solutions available. First, you could write a simple program on the micro to set the high order bit. Second, if the file is going from micro to Cyber to micro, and is not going to be used on the Cyber, it could be transferred as a micro binary.

4. In transferring text files or Cyber binaries, special characters are used to indicate file markers. For binaries, 375(8) indicates an EOR, 376(8) indicates an EOF, and 373(8) indicates an EOI. For text files, where end of line (EOL) is defined as carriage return (CR) or carriage return and line feed (CR LF), #EOR(EOL) indicates an EOR, #EOF(EOL) indicates an EOF, and #EOI(EOL) indicates an EOI. These characters are always inserted in files being transferred from Cyber to micro. When these characters are encountered as a file is received from the micro, they are translated into the appropriate file markers. To be recognized as file markers in a Cyber binary, the file marker character must occur where a block length is expected. The next character would then be another file marker or a block length. To be recognized in a text file, the file marker characters must occur immediately after end of line. Any data blocks received after the EOI character are ACK'd or NAK'd, but the data is ignored.

   These characters have no special meaning if the file is transferred as a micro binary.

5. The first block transmitted from/to Cyber is of the form:

   SOH+200B / file type / 255D-file type / 123D / zeros

   File type is:
   - 00 - Text
   - 04 - Micro binary
   - 08 - Cyber binary

   This block will be recognized by ASCII Express for the Apple and by XMODEM. If it is not explicitly supported by your micro's transfer program, it will be considered an illegal block and will be NAK'ed. If this block is NAK'ed three times, the transfer will proceed with the data. If the first block received is of this form, and XMODEM is in automatic file type detection mode, the file type in the block will be used to determine the type of transfer. If an illegal file type is found, micro binary is used. If XMODEM is in automatic mode, and no file type block is received, file type text is set.

6. When using XMODEM via TELENET, issue 'set?b0:33,63:0' when in TELENET command mode to set 8-bit ascii mode. This can be done before connecting to the host, or you can escape to TELENET command mode by typing (CR)@(CR). After typing the SET command, you can reenter the host with the CONT command.

Using XMODEM

Log into the Cyber using the appropriate micro communications package (such as ASCII Express on the APPLE). Make sure the line you are using will handle PARITY=NONE. All eight data bits must be passed.

Type:

XMODEM.

You will receive the prompt:

Host to Send (S) or Receive (R) a file?

Enter "S" (no quotes) to send a file from the Cyber to a micro. Enter "R" to receive a file from a micro. You may enter the entire word "SEND" or "RECEIVE".

SEND

If you entered "SEND", the next prompt will be:

Please enter the file name.

The file you specify must be a local file that can be read, and must not be empty. If the file is empty, the prompt for file name will be repeated. If the file you wish to transfer is permanent, you must do a GET or ATTACH before executing XMODEM.

XMODEM will then ask:

Is the file -
T = Text
B = Cyber binary
M = Micro binary

Enter the character that describes the type of file you are sending.

TEXT file

If you indicate the file is a text file, you will be asked:

```
Is the text on the Cyber -
A = 6/12 Ascii (upper/lower case)
C = CDC Display code (upper case only)
E = 8/12 Ascii character set
```

Enter the character that describes the character set of the file.

XMODEM will then print out the approximate number of blocks to be transfered.

At this point, you should enter the appropriate commands on the micro (CTRL-G, etc.) to initiate the file transfer. If XMODEM does not receive a NAK after 10 tries, it will abort with the message:

```
No initial NAK received.
```

Thus, if you decide to abort the transfer before it has started, enter 11 of any character except NAK (CTRL-U). If you decide to abort the transfer after it has started, enter a CTRL-X (ASCII CAN).

RECEIVE

If you entered "RECEIVE", the first prompt will be:

```
Please enter the file name.
```

Enter the file name to be used on the Cyber. It must be a valid NOS local file name, and must have WRITE access.

XMODEM will then ask:

```
Is the file -
T = Text
B = Cyber binary
M = Micro binary
S = Auto select
```

If you specified automatic file type detection, the first block received will determine the type of transfer (see note 6 above). If the file type detected is TEXT, 6/12 display code will be used for the character set.

If you indicated the file is a text file (option T), you will be asked:

Is the text on the Cyber -
A = 6/12 Ascii (upper/lower case)
C = CDC Display Code (upper case only)
E = 8/12 Ascii character set

This option determines in which character set the file will be stored.

If option C is selected, characters which cannot be translated to the 64 character set, except for NULL's, will be set to blanks. NULL's will be ignored. Lower case alphabetics will be translated to upper case.

If you decide to abort the transfer after it has started, enter a CTRL-D (ASCII EOT).

Command Parameters

Instead of using the menu mode described above, you may use command parameters. The available parameters are:

FN - Local File Name
TD - Transfer Direction (Send or Receive)
FT - File Type
  A = ASCII (6/12 on NOS) to/from ASCII (8 bit on micro)
  B = Cyber binary
  T = Text (uses current character set)
  C = CDC Display Code to/from ASCII (8 bit on micro)
  E = ASCII (8/12 on NOS) to/from ASCII (8 bit on micro)
  M = Micro binary
  S = Automatic Selection (receive only). If special control characters are transmitted (see note 6 above) but unrecognized, file type is assumed to be M. If control characters are not present, file type is assumed to be T.
LF - Line Feed.
  YES = End of line requires CR LF.
  NO = End of line indicated by CR (no LF needed)
  Default = YES.
  A parameter value beginning with Y or N will be accepted.

SP - Suppress special file type block.
NO = Do not suppress special file type block (default).
YES = Suppress special file type block.

Prompts will be issued for any illegal or missing parameters, except the LF parameter. Parameters may be specified in positional or keyword mode, but a positional may not follow a keyword parameter.

Examples:

```
XMODEM,FN=TEST,TM=SEND,FT=C,LF=YES.
```

would send the CDC Display Code text file TEST, maintaining file markers, with end of lines transferred as CR LF. An equivalent control statement would be:

```
XMODEM,TEST,S,C.
```

To receive file ABC using automatic file type detection, enter:

```
XMODEM,ABC,R,S.
```

To be prompted for FN, TD, FT and to select no line feeds at end of line, enter:

```
XMODEM,LF=N.
```

Appendix D: XMODEM User Guide for pre-NOS/VE 1.1.2 systems

XMODEM is a program for transferring files (text files, Cyber 180 binaries, and micro binaries) between a Cyber running VEIAF and a micro using Christensen protocol. File characteristics determine the type of file transfer from Cyber to micro, and are set when a file is transferred from micro to Cyber. Files are transferred as follows:

- Micro binaries. A micro binary is indicated when the USER_INFORMATION file attribute is set to 'MICRO_BINARY'. The file is read and written character for character. A Cyber file with UI=MICRO_BINARY is transferred in this manner regardless of other file attributes.

- Text. A text file is read and written a line at a time. If a Cyber file has the attribute RECORD_TYPE=variable (and doesn't have UI=MICRO_BINARY), it is transferred to the micro as a text file. An ASCII CR or CR LF is appended to the end of each line. When a text file is received from a micro, a carriage return is used to indicate the end of line. The file attribute RECORD_TYPE =VARIABLE is set when the Cyber file is opened. End of partitions are retained. An LF immediately after a CR is ignored.

- Cyber 180 binaries. A Cyber file being transferred to a micro is transferred as a Cyber binary if the file attribute RECORD TYPE =UNDEFINED is set. When a file is being received, RECORD_TYPE=UNDEFINED, FILE_STRUCTURE=DATA, and FILE_CONTENTS =OBJECT are set. The file is blocked as it is sent to the micro. The first character of each block is the block length (in characters). A block length of 373(8) indicates End-of-Information. This method allows the exact length of the file to be preserved.

- Cyber 180 object libraries are handled identically to 180 binaries, except FILE_STRUCTURE = LIBRARY is set when the file is received.

Several things must be considered when using XMODEM:

1. PARITY=NONE must be possible. Christensen protocol is an 8 bit protocol, and requires all 8 data bits to be passed. Data networks, such as some NET/ONE dialups, which are always set to even parity may not be used. XMODEM will automatically set PARITY=NONE before the transfer, and restore it before ending.

2. XMODEM must be used at VEIAF level 1.1.1 or above.

3. The first block transmitted by XMODEM is of the form:

   201(8) / file_type / 255D-file_type / 123D / zeroes (128 of them)

   The file type is:

   00 - Text
   04 - Micro binary
   08 - Cyber binary
   40(16) - Object library

   This is a nonstandard block. It is recognized by ASCII express for the Apple and by XMODEM. If it is not recognized by the receiver, it will be NAK'ed. After three NAK's, XMODEM will proceed with the file transfer. If XMODEM is in auto mode, and the first block received is of the above format, the file type will be used to set the type of transfer. If the file type is unrecognized, micro binary type will be assumed. If XMODEM is in auto mode and the first block is not of the above format, the file will be assumed to be a text file.

4. Some micros, such as the APPLE, require the high order bit in each character to be set for certain text files. If a file is transferred as a text file, it will not have the high order bit set. There are two solutions available. First, you could write a simple program on the micro to set the high order bit. Second, if the file is going from micro to Cyber to micro, and is not going to be used on the Cyber, it could be transferred as a micro binary.

5. In transferring text files or Cyber binaries, special characters are used to indicate file markers. For binaries, 375(8) indicates an EOR, 376(8) indicates an EOF, and 373(8) indicates an EOI. For text files, where end of line (EOL) is defined as carriage return (CR) or carriage return and line feed (CR LF), #EOR(EOL) indicates an EOR, #EOF(EOL) indicates an EOF, and #EOI(EOL) indicates an EOI. These characters are always inserted in files being transferred from Cyber to micro. When these characters are encountered as a file is received from the micro, they are translated into the appropriate file markers. To be recognized as file markers in a Cyber binary, the file marker character must occur where a block length is expected. The next character would then be another file marker or a block length. To be recognized in a text file, the file marker characters must occur immediately after end of line.

   These characters have no special meaning if the file is transferred as a micro binary.

6. Currently, due to a deficiency in transparent input, if a file does not have the special EOI character(s), transmisson will hang after the last block is transmitted from micro to Cyber. The last block transmitted from the micro is a single character, EOT. As the Cyber is expecting 132 characters, it cannot respond to the EOT. The micro will repeatedly timeout. When this happens, abort the micro transfer program and enter 131 or more blanks followed by a carriage return. This will fill out the last block. XMODEM will see that the first character of that block is an EOT and shut down. If the file contains the special EOI character (373(8) for binary file or #EOI (CR) for text file), and the file is not being transferred as a micro binary, this problem will not occur. When XMODEM sees the EOI character, it knows to expect only one character in the next odd numbered block.

   Files transferred to the micro as a Cyber binary or text file always have the special EOI characters.

7. When receiving a file from the micro with XMODEM, an extraneous error message will sometimes occur after the transfer is complete when using some micros (such as Apple). This is because some micros always send an even number of blocks, so there may be an extraneous block after the EOI character. The message 'Expecting command, found ???...???' may occur on NOS/VE. This problem can be fixed when DAP S4657 is implemented.

8. When using XMODEM via TELENET, issue 'set?b0:33,63:0' when in TELENET command mode to set 8-bit asciimode. This can be done before connecting to the host, or you can escape to TELENET command mode by typing (CR)@(CR). After typing the SET command, you can reenter the host with the CONT command.

Using XMODEM

To initiate XMODEM, enter:

    XMODEM [,parameters]

XMODEM has 4 parameters - FILE_NAME (FN), TRANSFER_DIRECTION (TD), FILE_TYPE (FT) and LINE_FEED (LF). None are required.

FILE_NAME may be any valid file name or file path.

TRANSFER_DIRECTION may be set to SEND (S) or RECEIVE (R). If TRANSFER_DIRECTION is omitted, you will be prompted for SEND or RECEIVE.

FILE_TYPE may be set to B (Cyber binary), M (Micro binary), O (Object library), T (Text), or S (Auto select) when TD=R (if TD=S, file type parameter is ignored and is not prompted for).

LINE_FEED may be YES/TRUE/ON to indicate a CR LF is needed for end of line or NO/FALSE/OFF to indicate just CR signals end of line. Default=YES.

SEND

If you select the SEND option, and FILE_NAME was not specified, you will be prompted:

Please enter the file name.

Enter the name of the Cyber file you wish to send. It may be any valid file path (e.g. ABC or $USER.ABC). The file must be available with read access, and must not be empty. If the file is empty or cannot be read, you will be prompted again for the file name. If FILE_NAME was specified on the control card, you will not be prompted for the file name unless you entered the name of a file which cannot be read.

When a valid file name has been entered, you will receive a message indicating the number of blocks to be transferred.

At this point, you should enter the appropriate commands on the micro to place it in receive mode. The transfer will start when the micro sends a NAK to the Cyber.

If for any reason you wish to abort the transfer after this point, enter a CTRL-X (ASCII CAN).

RECEIVE

If FILE_NAME was not specified on the control card, the first prompt after selecting RECEIVE mode will also be:

Please enter the file name.

Any valid file path may be specified. If the specified file does not exist, it will be created. If the file already contains data, it must have MODIFY, SHORTEN, and APPEND access. If the file is empty, it must have APPEND permission. If the file is a new file (i.e. it has never been opened), there are no further restrictions.

If it is an old file (i.e it has been opened, regardless of whether or not it contains data), the record type must be compatible with the type of file being received. If the file is binary or micro binary, RECORD_TYPE=UNDEFINED must be set. If the file is text, RECORD_TYPE=VARIABLE must be set.

If FILE_NAME was specified on the control card, you will not be prompted for it unless an invalid file name was entered.

The next prompt will be:

```
Is the file -
   T = Text file
   B = Cyber binary
   O = Object library
   M = Micro binary
   S = Auto select
```

Enter the character of the type of file you will be receiving. If you enter option S, the file type will be determined by the first block received (see note 3 above). It is at this point that XMODEM guarantees that the record types are compatible if the file is an old file. If they are not compatible, you will be prompted for a new file name. However, the compatibility cannot be verified if auto select is used. If you are writing on an old file, and use auto select, be sure that the record types are compatible before hand. If they are not, XMODEM will abort the first time it tries to write on the old file.

Next, you should enter the appropriate commands on the micro to place the micro in SEND mode. On early NOS/VE systems, the micro must be ready to send within twenty seconds. As timed input is not yet available, XMODEM can only issue the required initial NAK once. It then goes into input mode waiting for a response. If no response is received, the micro and Cyber are in a deadlock. The micro requires a NAK to initiate the transfer, but the Cyber is in input mode waiting for a block of data. XMODEM cannot regain control after it enters input mode. If this deadlock does occur, enter 132 or more CTRL-D's. You may hold CTRL-D and REPEAT down until the program quits and you receive the slash (/) prompt. This is necessary because each block must be 132 characters. If the first character in a block is an ASCII EOT (CTRL-D), XMODEM will quit. This problem can be fixed when DAP S4657 is implemented.

Appendix E: XMODEM User Guide for NOS/VE 1.1.2 system

XMODEM is a program for transferring files (text files, Cyber 180 binaries, and micro binaries) between a Cyber running VEIAF and a micro using Christensen protocol. File characteristics determine the type of file transfer from Cyber to micro, and are set when a file is transferred from micro to Cyber. Files are transferred as follows:

- Micro binaries. A micro binary is indicated when the USER_INFORMATION file attribute is set to 'MICRO_BINARY'. The file is read and written character for character. A Cyber file with UI=MICRO_BINARY is transferred in this manner regardless of other file attributes.

- Text. A text file is read and written a line at a time. If a Cyber file has the attribute RECORD_TYPE=variable (and doesn't have UI=MICRO_BINARY), it is transferred to the micro as a text file. An ASCII CR or CR LF is appended to the end of each line. When a text file is received from a micro, a carriage return is used to indicate the end of line. The file attribute RECORD_TYPE =VARIABLE is set when the Cyber file is opened. End of partitions are retained. An LF immediately after a CR is ignored.

- Cyber 180 binaries. A Cyber file being transferred to a micro is transferred as a Cyber binary if the file attribute RECORD_TYPE =UNDEFINED is set. When a file is being received, RECORD_TYPE=UNDEFINED, FILE_STRUCTURE=DATA, and FILE_CONTENTS =OBJECT are set. The file is blocked as it is sent to the micro. The first character of each block is the block length (in characters). A block length of 373(8) indicates End-of-Information. This method allows the exact length of the file to be preserved.

- Cyber 180 object libraries are handled identically to 180 binaries, except FILE_STRUCTURE = LIBRARY is set when the file is received.

Several things must be considered when using XMODEM:

1. PARITY=NONE must be possible. Christensen protocol is an 8 bit protocol, and requires all 8 data bits to be passed. Data networks, such as some NET/ONE dialups, which are always set to even parity may not be used. XMODEM will automatically set PARITY=NONE before the transfer, and restore it before ending.

2. The first block transmitted by XMODEM is of the form:

   201(8) / file_type / 255D-file_type / 123D / zeroes (128 of them)

   The file type is:

   00 - Text
   04 - Micro binary
   08 - Cyber binary
   40(16) - Object library

   This is a nonstandard block. It is recognized by ASCII express for the Apple and by XMODEM. If it is not recognized by the receiver, it will be NAK'ed. After three NAK's, XMODEM will proceed with the file transfer. If XMODEM is in auto mode, and the first block received is of the above format, the file type will be used to set the type of transfer. If the file type is unrecognized, micro binary type will be assumed. If XMODEM is in auto mode and the first block is not of the above format, the file will be assumed to be a text file.

3. Some micros, such as the APPLE, require the high order bit in each character to be set for certain text files. If a file is transferred as a text file, it will not have the high order bit set. There are two solutions available. First, you could write a simple program on the micro to set the high order bit. Second, if the file is going from micro to Cyber to micro, and is not going to be used on the Cyber, it could be transferred as a micro binary.

4. In transferring text files or Cyber binaries, special characters are used to indicate file markers. For binaries, 375(8) indicates an EOR, 376(8) indicates an EOF, and 373(8) indicates an EOI. For text files, where end of line (EOL) is defined as carriage return (CR) or carriage return and line feed (CR LF), #EOR(EOL) indicates an EOR, #EOF(EOL) indicates an EOF, and #EOI(EOL) indicates an EOI. These characters are always inserted in files being transferred from Cyber to micro. When these characters are encountered as a file is received from the micro, they are translated into the appropriate file markers. To be recognized as file markers in a Cyber binary, the file marker character must occur where a block length is expected. The next character would then be another file marker or a block length. To be recognized in a text file, the file marker characters must occur immediately after end of line. Any data blocks received after the EOI character are ACK'd or NAK'd, but the data is ignored.

   These characters have no special meaning if the file is transferred as a micro binary.

5. When using XMODEM via TELENET, issue 'set?b0:33,63:0' when in TELENET command mode to set 8-bit asciimode. This can be done before connecting to the host, or you can escape to TELENET command mode by typing (CR)@(CR). After typing the SET command, you can reenter the host with the CONT command.

Using XMODEM

To initiate XMODEM, enter:

XMODEM [,parameters]

XMODEM has 4 parameters - FILE_NAME (FN), TRANSFER_DIRECTION (TD), FILE_TYPE (FT) and LINE_FEED (LF). None are required.

FILE_NAME may be any valid file name or file path.

TRANSFER_DIRECTION may be set to SEND (S) or RECEIVE (R). If TRANSFER_DIRECTION is omitted, you will be prompted for SEND or RECEIVE.

FILE_TYPE may be set to B (Cyber binary), M (Micro binary), O (Object library), T (Text), or S (Auto select) when TD=R (if TD=S, file type parameter is ignored and is not prompted for).

LINE_FEED may be YES/TRUE/ON to indicate a CR LF is needed for end of line or NO/FALSE/OFF to indicate just CR signals end of line. Default=YES.

SEND

If you select the SEND option, and FILE_NAME was not specified, you will be prompted:

Please enter the file name.

Enter the name of the Cyber file you wish to send. It may be any valid file path (e.g. ABC or $USER.ABC). The file must be available with read access, and must not be empty. If the file is empty or cannot be read, you will be prompted again for the file name. If FILE_NAME was specified on the control card, you will not be prompted for the file name unless you entered the name of a file which cannot be read.

When a valid file name has been entered, you will receive a message indicating the number of blocks to be transferred.

At this point, you should enter the appropriate commands on the micro to place it in receive mode. The transfer will start when the micro sends a NAK to the Cyber.

If for any reason you wish to abort the transfer after this point, enter a CTRL-X (ASCII CAN).

RECEIVE

If FILE_NAME was not specified on the control card, the first prompt after selecting RECEIVE mode will also be:

Please enter the file name.

Any valid file path may be specified. If the specified file does not exist, it will be created. If the file already contains data, it must have MODIFY, SHORTEN, and APPEND access. If the file is empty, it must have APPEND permission. If the file is a new file (i.e. it has never been opened), there are no further restrictions.

If it is an old file (i.e it has been opened, regardless of whether or not it contains data), the record type must be compatible with the type of file being received. If the file is binary or micro binary, RECORD_TYPE=UNDEFINED must be set. If the file is text, RECORD_TYPE=VARIABLE must be set.

If FILE_NAME was specified on the control card, you will not be prompted for it unless an invalid file name was entered.

The next prompt will be:

Is the file -
T = Text file
B = Cyber binary
O = Object library
M = Micro binary
S = Auto select

Enter the character of the type of file you will be receiving. If you enter option S, the file type will be determined by the first block received (see note 2 above). It is at this point that XMODEM guarantees that the record types are compatible if the file is an old file. If they are not compatible, you will be prompted for a new file name. However, the compatibility cannot be verified if auto select is used. If you are writing on an old file, and use auto select, be sure that the record types are compatible before hand. If they are not, XMODEM will abort the first time it tries to write on the old file.

Next, you should enter the appropriate commands on the micro to place the micro in SEND mode. On early NOS/VE systems, the micro must be ready to send within twenty seconds. As timed input is not yet available, XMODEM can only issue the required initial NAK once. It then goes into input mode waiting for a response. If no response is received, the micro and Cyber are in a deadlock. The micro requires a NAK to initiate the transfer, but the Cyber is in input mode waiting for a block of data. XMODEM cannot regain control after it enters input mode. If this deadlock does occur, enter a CTRL-D (ASCII EOT).