TM-70-1031-2

TECHNICAL MEMORANDUM

BCMIO - THE BELLCOMM EXEC 8 FORTRAN I/O LIBRARY

Bellcomm

BELLCOMM, INC.
955 L'ENFANT PLAZA NORTH, S.W., WASHINGTON, D.C. 20024

COVER SHEET FOR TECHNICAL MEMORANDUM

TITLE- BCMIO - The Bellcomm Exec 8 FORTRAN I/O Library

TM- 70-1031-2

DATE- September 15, 1970

FILING CASE NO(S)- 803

AUTHOR(S)- W. M. Keese

FILING SUBJECT(S) (ASSIGNED BY AUTHOR(S))- FORTRAN, Input, Output, I/O

ABSTRACT

Efficiency of the UNIVAC 1108 computer is frequently controlled more by the programming of Input/Output than it is by the programming of the computations. Programs written in the FORTRAN language (as most of Bellcomm's are) effect their Input/Output through implicit use of a set of subroutines called the FORTRAN I/O library.

A new Exec 8 library, BCMIO, has been programmed at Bellcomm to conform to the standard format established by the UNIVAC Standard Exchange (USE). In order to use the library, it is necessary to convert data files to the BCMIO format, but program alterations are not required. This library has been part of the regular operating system at Bellcomm since December, 1969. It is now undergoing checkout at two other installations, prior to general release.

BCMIO reduces program execution times by 20-85%. Taking 50% as the midpoint of this range, and assuming that half of the machine time is devoted to program execution there is a UNIVAC 1108 throughput increase of 33% resulting from use of the new library.

SEE REVERSE SIDE FOR DISTRIBUTION LIST

BELLCOMM, INC.

TM-70-1031-2

DISTRIBUTION

COMPLETE MEMORANDUM TO

CORRESPONDENCE FILES:

OFFICIAL FILE COPY
plus one white copy for each
additional case referenced

TECHNICAL LIBRARY (4)

NASA

J. Costantino/MRT
K. Joseph/TN
A. S. Lyman/MR

Goddard Spaceflight Center

O. Jones/565

Jet Propulsion Lab

B. Jirka

Manned Spacecraft Center

R. Everett/ED

MSFC

J. Lynn/ S&E-COMP-DS

Slidell

E. Brooks/PM-SAT-S-1B/1

University of Maryland

J. Menard

Bellcomm, Inc.

G. M. Anderson
A. P. Boysen, Jr.
J. O. Cappellari, Jr.
K. R. Carpenter
C. L. Davis
D. A. DeGraaf
J. P. Downs
D. R. Hagner
W. G. Heffron
H. A. Helm
J. J. Hibbert
H. S. London
K. E. Martersteck
J. Z. Menard
J. M. Nervik
N. W. Hinners
G. T. Orrok
I. M. Ross
P. F. Sennewald
R. V. Sperry
W. Strack
W. B. Thompson
J. W. Timko
R. L. Wagner
M. P. Wilson
All Members, Department 1015
All Members, Division 103

BELLCOMM, INC.
955 L'ENFANT PLAZA NORTH, S.W. WASHINGTON, D. C. 20024

SUBJECT: BCMIO - The Bellcomm Exec 8 FORTRAN I/O Library - Case 803

DATE: September 15, 1970

FROM: W. M. Keese

TM-70-1031-2

TECHNICAL MEMORANDUM

Introduction

On a computer as fast as the UNIVAC 1108, most programs are I/O (Input/Output) bound. That is, computations can be done in less time than that required to read in the data and print out the results. Thus, the efficiency of computer utilization may depend more on the I/O process than on the computation algorithms.

Most programs at Bellcomm are written in the FORTRAN language. Such programs do their I/O through a set of library subroutines, collectively called the FORTRAN I/O library. This library has been rewritten at Bellcomm in order to obtain greater efficiency from the computer and mass storage devices. The result, called BCMIO, is now being used successfully at Bellcomm. Utilizing record packing and buffering, it reduces program execution times by 20-85%. Capabilities, features and characteristics of BCMIO are described in this memorandum.

1. Capabilities

BCMIO does not require any changes to any FORTRAN programs. It meets full ANSI FORTRAN standards. It meets all UNIVAC specifications with the exception of: 1) alternate symbiont support; 2) direct setting of random access addresses; and 3) user error control in certain "delayed error" situations. In addition, it provides some new facilities: most notably the support of multireel tape files.

In the following details, "Mass storage" refers to word-addressable drum, FASTRAND, and magnetic tape (including multireel files). "Symbionts" refer to unit record devices such as card readers, printers, card punches, remote terminals, teletypewriters, etc.

1.1 Detailed Support

BCMIO supports:

a. binary reads and writes (FORTRAN II, IV, & V) on mass storage;

b. formatted reads and writes (FORTRAN II, IV, & V) on mass storage and symbionts;

c. ENDFILE on mass storage;

d. BACKSPACE on mass storage (not allowed over reel boundaries on multireel files);

e. REWIND on mass storage (to start of 1st reel on multireel files);

f. NAMELIST reads and writes on mass storage and symbionts;

g. CLOSE on mass storage and symbionts;

h. END and ERR control for all I/O calls in which the FORTRAN compiler will pass these to the library. (BCMIO will give user control on these conditions only if they arise before control has been normally returned.)

1.2 UNIVAC Features not Supported

a. BCMIO does not support alternate symbionts. It deals with files in the USE standard SDF format (sec. 2) in which there are no separate alternate symbiont files. (However, inclusion of this feature would, in fact, be easy.)

b. BCMIO does not support the UNIVAC routine SETADR. Although implementation of this would be trivial, the feature is felt to be too dangerous for general use in an environment of files with control words.

c. There is also a restriction on user control in the case of END or other ERRor conditions; some errors are not detected until after normal control has returned to the user. (Indeed, it is possible for several delayed errors to reveal themselves all at once!) In case of such ambiguity, BCMIO will error the execution with appropriate diagnostics.

1.3 New Features in BCMIO

a. BCMIO supports multireel tape files.

b. Elements LRCIN and LRCOT [A,B]* provide reentrant interfaces to BCMIO for multiprogramming FORTRAN activities. Further, LRCIN permits reading FORTRAN records of unknown length.

c. POSEND [C] positions a file for continuation, regardless of how it and/or its creation was previously terminated.

d. BCMIO distinguishes between batch and demand runs for diagnostic purposes. As a spin-off from this, several functions regarding the run are available in element DEMAND [D].

e. Alternate calls to TRCFIO [E] turn on and off a trace mode on all non-symbiont I/O.

2. USE Format - Tape Conversion

BCMIO implements the FORTRAN part of the USE standard on I/O formats [1]. The format was designed with the primary objectives of: 1) across-the-board interprocessor compatibility within Exec 8; 2) I/O efficiency; and 3) ability to recover data past errors. A minor design consideration was deliberate incompatibility with past Exec 8 tapes. (It is cheaper to convert tapes than to carry the programming overhead for compatibility.)

Thus, it is necessary to convert all old files to the USE format in order to use BCMIO. (However, the USE format is so much more efficient for most files, that the conversion is paid for within 1 or 2 uses of the converted file.)

A conversion routine, CNVTFT, is included in BCMIO [F]. This converts files according to data card direction. It will accept Exec 2 FORTRAN tapes, 7090 FORTRAN tapes, and Exec 8 FORTRAN binary tapes as input, producing USE tapes as output.

3. Techniques

BCMIO achieves efficiency of the CPU and mass storage devices by packing logical records together into fixed size physical records. This minimizes the amount of real physical I/O.

*Bracketed letters indicate appendices; bracketed numbers indicate references.

This packing is done in a pool of buffers, each of which can be attached to any logical unit as needed. The Bellcomm version has 10 buffers in the pool, which makes BCMIO the same size as the smallest UNIVAC library. This number can be altered by changing only one card in element BFFERS. It is desirable that there be at least as many buffers as there are logical units active at a given time. (Special users can have their own versions of BFFERS.)

BCMIO does its I/O asynchronously, concurrently with computation. It is interrupt driven, completely self contained, and requires no changes to the executive or to the programs using it.

4. Relative Efficiency

"Efficiency" is difficult to define in any overall sense in an environment such as Exec 8 where various programs are sharing the computer facilities at any given moment. Further, the gathering of data for a statistically adequate measurement of the effects of an I/O package is unduly expensive even in a simpler environment. For instance, although buffering has long been regarded as highly important to computer efficiency, it should not be important in a multi-programming machine; the machine should be able to execute computations for one program while another program is doing I/O. However, practical considerations indicate that buffering is still almost as important as ever. BCMIO is buffered, but buffering benefits are not considered below.

The total size of BCMIO (as distributed with 10 buffers) runs about the same as the size of the smallest UNIVAC package, so memory requirements are not changed.

Figure 1 shows FASTRAND (or drum) packing fractions for each of the two packages. In general, BCMIO serves as an envelope to the best points of the discontinuous UNIVAC graph. For the average length* binary record of 31 words, BCMIO puts nearly twice as much data on the same amount of FASTRAND. Figure 2 shows the same data for magnetic tape. Here, BCMIO exceeds by a factor of 3 for the average record size, but does not do as well in some selected regions.

*According to an unpublished Sandia study.

Efficiency with respect to speed varies with how often the device is accessed. It is a function of logical record length and of the distribution in time of the accesses. Ignoring the benefits which BCMIO derives from buffering, this can be reduced to a number of accesses situation. Figure 3 shows number of data words per access as a function of record length. At the average length of 31 words, BCMIO shows a 7 to 1 advantage. However, as record length increases the UNIVAC peaks will show broader regions above the BCMIO curve. Somewhere above 2000 words UNIVAC will consistantly be better, approaching 13% above BCMIO. This is entirely due to choice of block size. The BCMIO figure was chosen for compatibility with other UNIVAC processors, rather than speed.

In practice, BCMIO has proved 6 to 7 times as good as UNIVAC on heavily I/O bound programs (in startling agreement with the Sandia figure for average record length). It shows approximately a 20% improvement on BCMASP, our largest compute-bound run. These figures are based on the Bellcomm charge algorithm [2], a weighted sum of CPU, core, and I/O usage.

This charge algorithm does not reflect any advantage for overlapped I/O, so the total benefit to the installation probably is better than this.

Taking the middle of this 20-85% improvement, and assuming that half of our computer time is spent in execution, this indicates approximately a 25% reduction in total computer system requirements, a 33% throughput increase.

1031-WMK-cp

W. M. Keese

Attachments
Figures 1 - 3
Appendix A - F

REFERENCES

1. Minutes of UNIVAC Scientific Exchange, March 24-29, 1969, p. 109.

2. "Accounting of Computer System Use in Exec 8", A. L. Rothstein, Bellcomm Memorandum for File, June 5, 1969.

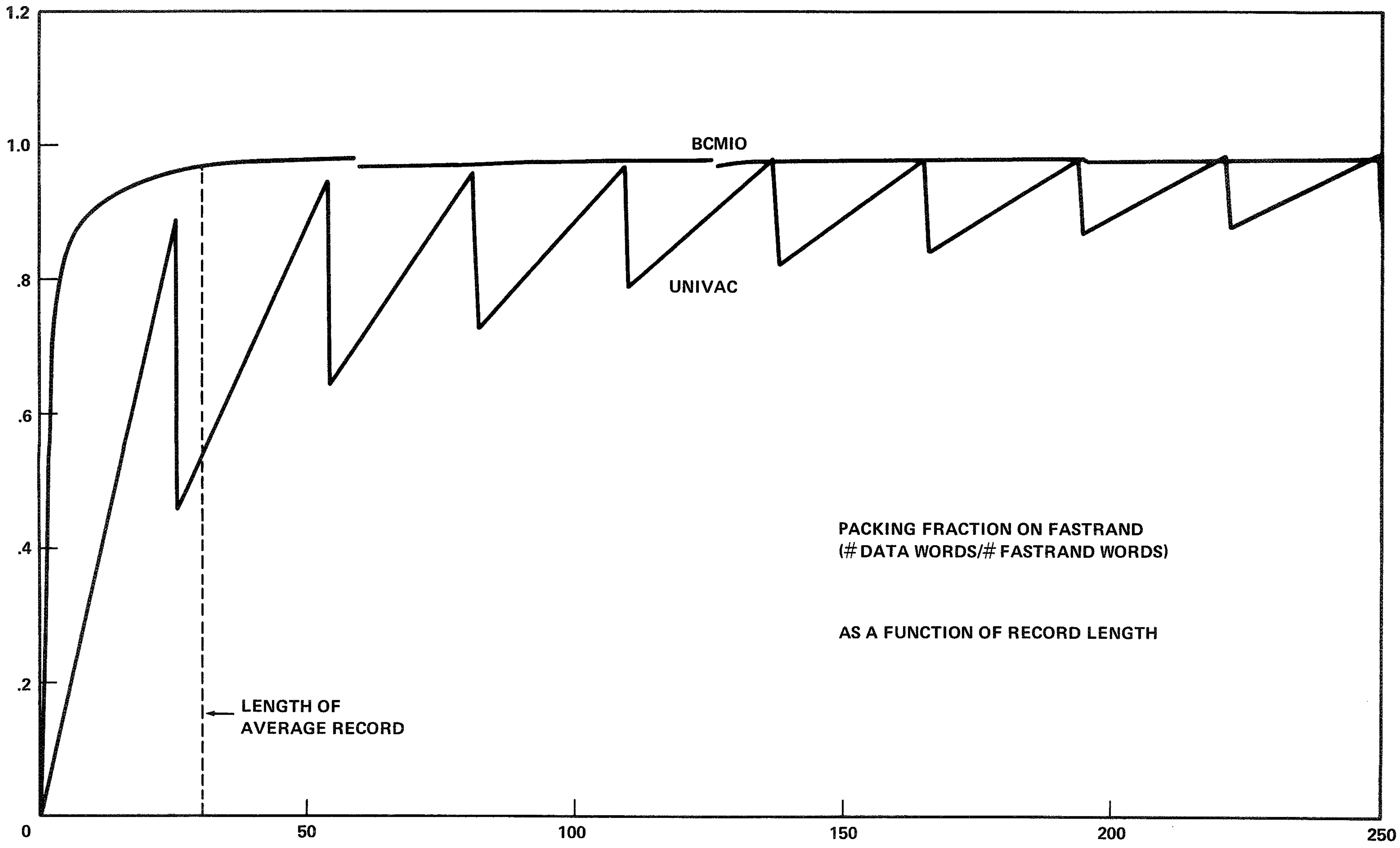
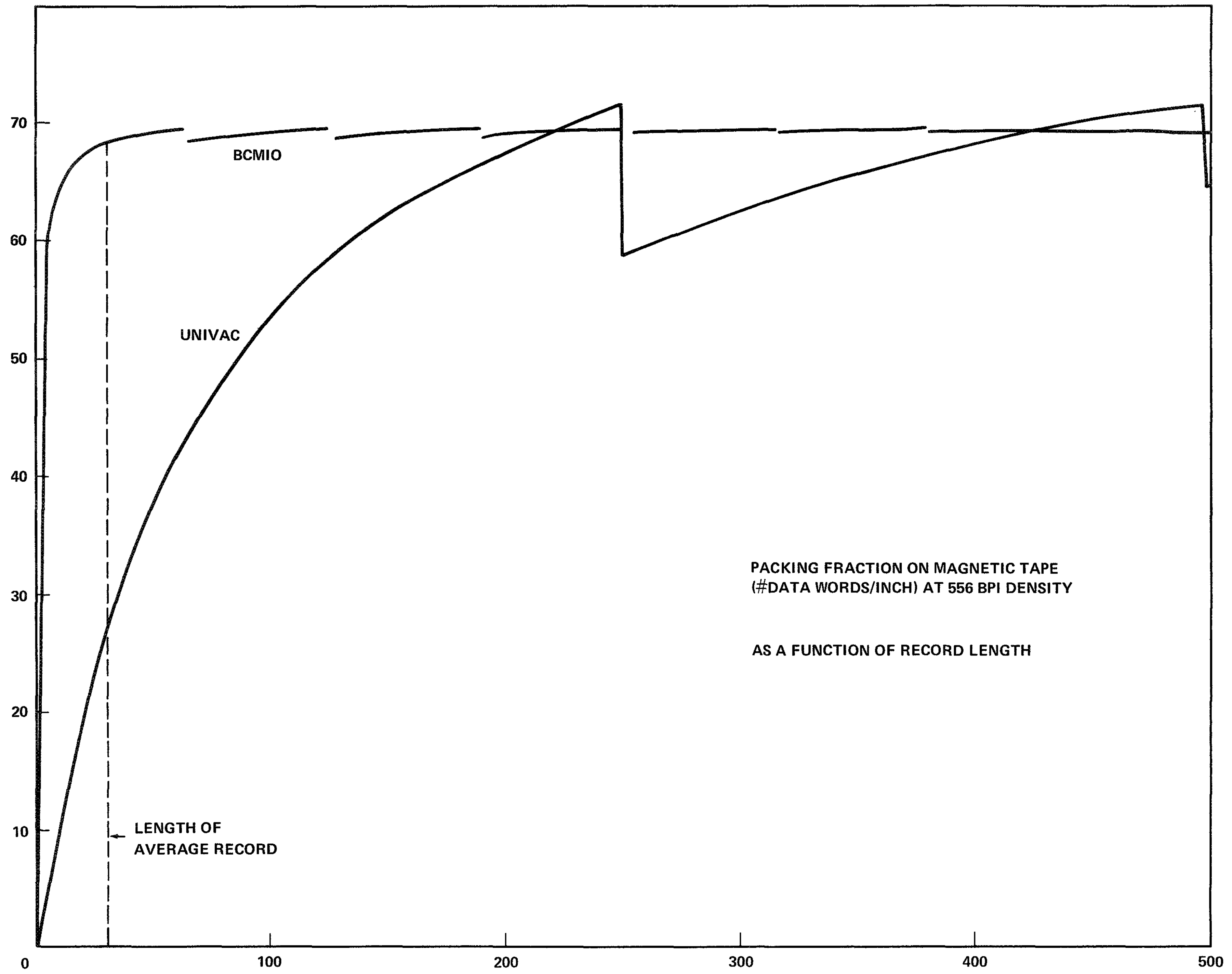

1.2
1.0
.8
.6
.4
.2
0
50
100
150
200
250
BCMIO
UNIVAC
LENGTH OF
AVERAGE RECORD
PACKING FRACTION ON FASTRAND
(# DATA WORDS/# FASTRAND WORDS)
AS A FUNCTION OF RECORD LENGTH

FIGURE 1



PACKING FRACTION ON MAGNETIC TAPE
(#DATA WORDS/INCH) AT 556 BPI DENSITY

AS A FUNCTION OF RECORD LENGTH

FIGURE 2

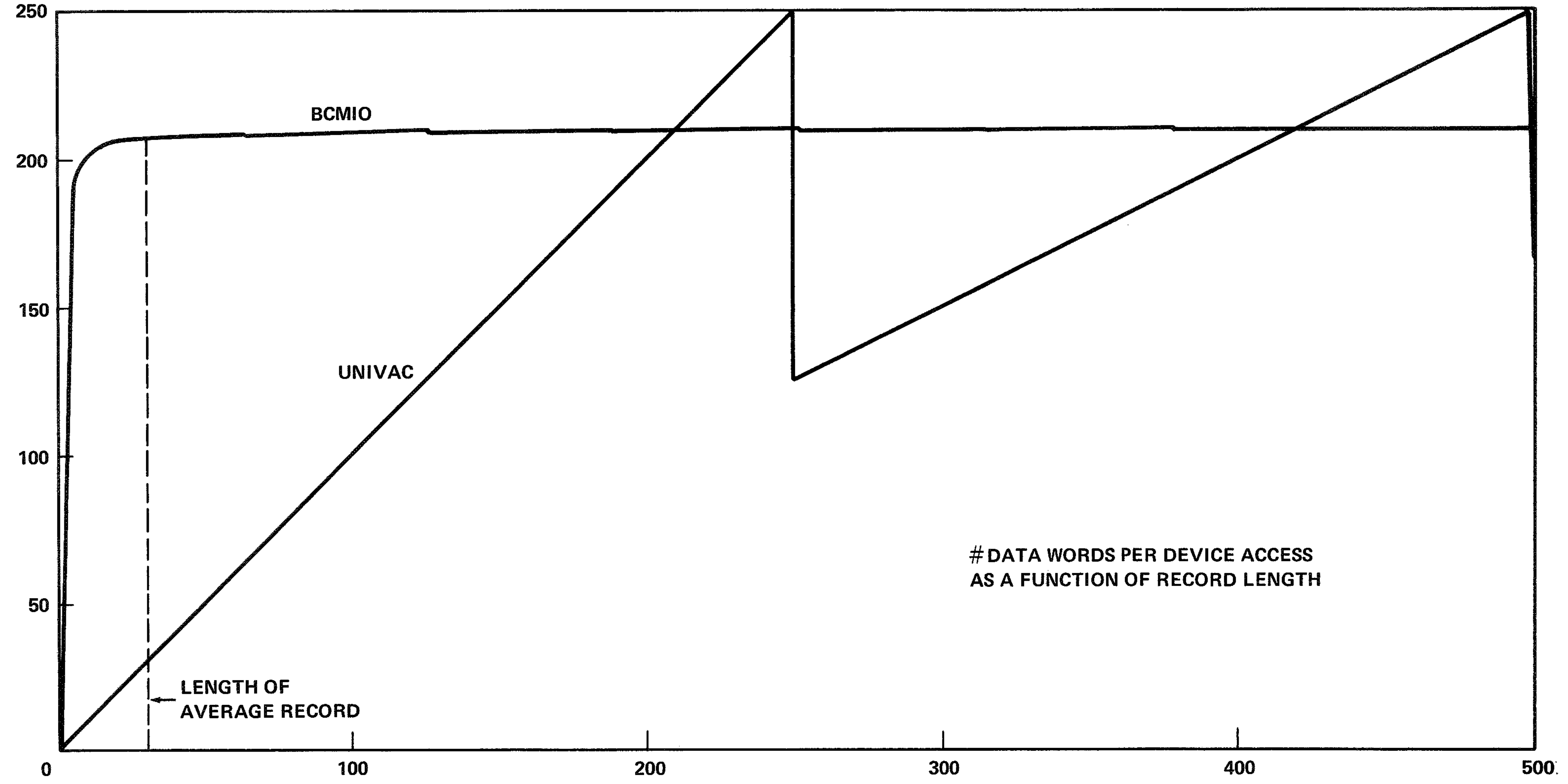

250
200
150
100
50
0
100
200
300
400
500
BCMIO
UNIVAC
LENGTH OF
AVERAGE RECORD
#DATA WORDS PER DEVICE ACCESS
AS A FUNCTION OF RECORD LENGTH

FIGURE 3

Appendix A

Function NLRCIN

Purpose   To provide reentrant input for FØRTRAN programs, and to provide input without knowledge of record length.

Call   NGØT=NLRCIN(NUNIT,NWANTD,WHERE,$EØF,$ERR)

where   (letter Ø, not digit 0)
NGØT will be set to the number words read,
NUNIT = the unit used,
NWANTD = the maximum number of words wanted,
WHERE is the name of the array to receive the input,
EØF is the statement number to receive control if an end is read, and
ERR is the statement number to receive control on an error.

Effect   Up to NWANTD words are read from unit NUNIT into array WHERE. NGØT is set to the number actually read.

Note   NLRCIN provides the most efficient form of input for FØRTRAN programs.

Note   If NUNIT is a symbiont, then it is an error, and control will go to ERR.

Appendix B

Subroutine NLRCØT

Purpose To provide reentrant output for FØRTRAN programs.

Call CALL NLRCØT(NUNIT,NWØRDS,WHERE)

where (letter Ø, not digit 0)
NUNIT = the FØRTRAN unit number to use,
NWØRDS = the number of words to output, and
WHERE is the name of the array containing the words to output.

Effect NWØRDS from. WHERE are output onto unit number NUNIT.

Note NLRCIN provides the most efficient form of FØRTRAN output.

Note If NUNIT is a symbiont, then it is an error.

Appendix C

Subroutine PØSEND

Purpose To so position a unit as to be able to continue output from where it was last written.

Call Call PØSEND(NUNIT)

where (letter Ø, not digit 0)
NUNIT = the number of the unit to be positioned.

Effect Unit NUNIT is positioned to just before either an end of file, error, or the end of information. It is left positioned there.

(Use of END and ERR control is not adequate to achieve this purpose, as these do not distinguish between end of file and end of information for the user, but leave the position past on end of file but in front of the end of information.)

Appendix D

Element DEMAND

Purpose This element provides functions which enable FØRTRAN programs to get at load time parameters from the RUN and XQT cards.

Entries DEMAND,MNDYYR,NØPTS,ØPTION.

Call BØØL=DEMAND(any number of dummy args)

Value True if and only if demand run.

Call NHØLLR=MNDYYR (any number of dummy args)

Value 6 Hollerith characters: 1st pair=month, 2nd=day, 3rd=year.

Call N=NØPTS(any number of dummy args)
where (letter Ø, not digit 0)

Value N=option word bit 10=1 ↔ A option on XQT
bit 35=1 ↔ Ƶ option on XQT
(bit 0=sign, bit 35=[binary] 1)

Call BØØL=ØPTIØN(any number of arguments)
(letter Ø, not digit 0)

each argument is Hollerith, consisting only of letters.

Value Each argument is regarded as a Boolean product. The value of the function is the Boolean sum of these products. Thus,
ØPTION('ABC','DE','F') would be true if
1) the XQT card contained options A, B, and C; or
2) the XQT card contained options D and E; or
3) the XQT card contained option F.
It would be false if and only if all of them [1-3] failed.

Appendix E

Subroutine TRCFIØ

Purpose To provide a trace on (non-symbiont) FØRTRAN I/Ø. (Originally intended to debug BCMIØ, this has proved valuable in debugging user runs.)

Call CALL TRCFIØ
(letter Ø, not digit 0)

Effect The first call turns on trace mode. The second turns it off. In general, each call reverses the current situation.

Note When the trace mode is on, each I/Ø call produces about 1-1/2 pages of output. This consists of a dump of the unit control packets and buffer heads (defined in FIØDEF/PRØC).

Appendix F

Program CNVTFT

Purpose Conversion of FØRTRAN data tapes.

Operation The program is driven by (any number of) data cards. Each of these contains an input unit number (anywhere) in cols. 8 through 13 and an output unit number (anywere) in cols. 16 through 21. Each such card directs the conversion of 1 file (with an ENDFILE).

Method CNVTFT does neither input nor output aside from its data control cards and a FØRTRAN ENDFILE on the output tape. Input is done by element GETFRC, and output is done by element PUTFRC. These two elements could each use FØRTRAN I/Ø and be given separate R-option collections with their appropriate FØRTRAN I/Ø libraries. In the version distributed with BCMIØ, GETFRC uses assembly language input and reasonably elaborate internal logic to decide what type of input tape it is reading; PUTFRC uses shortcuts into BCMIØ in order to save program size.

Capabilities The BCMIØ version of CNVTFT accepts all Exec II FØRTRAN data tapes, all 7090 FØRTRAN DATA tapes, and Exec 8 FØRTRAN binary data tapes as input. It does not accept Exec 8 FØRTRAN formatted tapes. Subroutine GETFRC (particularly its internal subroutine, NEWITM) would have to be rewritten to change the acceptable input set.

Note There is no limit on record length.