TECHNICAL REPORT RG-CR-80-8

INTERACTIVE TEXT EDITING FOR
THE SIGMA 5
FINAL REPORT

Kevin Counsellman
Science Applications, Inc.
2109 W. Clinton Av.
Huntsville, Al. 35805

For
Guidance and Control Directorate
U. S. Army Missile Laboratory

SEPTEMBER 1980

U.S. ARMY MISSILE COMMAND
Redstone Arsenal, Alabama 35898

THIS DOCUMENT IS BEST QUALITY PRACTICABLE.
THE COPY FURNISHED TO DDC CONTAINED A
SIGNIFICANT NUMBER OF PAGES WHICH DO NOT
REPRODUCE LEGIBLY.

APPROVED FOR PUBLIC RELEASE, DISTRIBUTION UNLIMITED
DISPOSITION INSTRUCTIONS

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED. DO NOT RETURN IT TO THE ORIGINATOR.

DISCLAIMER

THE FINDINGS IN THIS REPORT ARE NOT TO BE CONSTRUED AS AN OFFICIAL DEPARTMENT OF THE ARMY POSITION UNLESS SO DESIGNATED BY OTHER AUTHORIZED DOCUMENTS.

TRADE NAMES

USE OF TRADE NAMES OR MANUFACTURERS IN THIS REPORT DOES NOT CONSTITUTE AN OFFICIAL ENDORSEMENT OR APPROVAL OF THE USE OF SUCH COMMERCIAL HARDWARE OR SOFTWARE.
DISCLAIMER NOTICE

THIS DOCUMENT IS BEST QUALITY PRACTICABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.
This report examines the requirements and implementation of an interactive text editing facility for the Sigma 5.
TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>SECTION</th>
<th>CONTENT</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>II</td>
<td>EDITOR DESIGN AND IMPLEMENTATION</td>
<td>2</td>
</tr>
<tr>
<td>III</td>
<td>CONCLUSIONS AND RECOMMENDATIONS</td>
<td>7</td>
</tr>
<tr>
<td>IV</td>
<td>REFERENCES</td>
<td>8</td>
</tr>
<tr>
<td>V</td>
<td>APPENDIX 1</td>
<td>9</td>
</tr>
<tr>
<td>VI</td>
<td>APPENDIX 2</td>
<td>14</td>
</tr>
</tbody>
</table>
# LIST OF TABLES

<table>
<thead>
<tr>
<th>TABLE</th>
<th>PAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
</tbody>
</table>

1. **EDITOR COMMANDS**
2. **SAMPLE JOB STREAM**
INTRODUCTIO

In the early 1960s, when systems such as the Sigma 5 were being released, most software development was done in a batch processing environment. Facilities for the interactive development of software were the exception rather than the rule.

Steadily declining hardware costs coupled with rising software development costs have created a new emphasis on improving the efficiency of software development utilities through interactive processing.

This report examines the requirements and implementation of one such utility, an interactive text editor for the Sigma 5.
II. EDITOR DESIGN AND IMPLEMENTATION

In general, a text editor reads symbolic information (be it source code, text or whatever) from some input file into an area of memory, performs some operation(s) on this information, and writes the result to an output file. A more detailed discussion of this process follows.

Input and Output Files. An editor input or output file is defined as a file, residing on magnetic disk, containing zero or more lines. A line is an 80-byte long record, corresponding to the standard 80 column punched card.

For the Sigma 5, under the RBM operating system, these requirements imply a compressed file format. The FSIZE parameter of the output file should be set to the expected number of lines in the workspace buffer at the conclusion of an edit session.

Consideration was given to using dynamic mass storage allocation to allow the size of the output file to vary during an edit session. However, since RBM requires a contiguous area for all disk files, there is no way to guarantee that sufficient disk space exists at the end of an arbitrary output file for dynamic expansion.

Workspace Buffer. In order to efficiently process the data read from the input file, this data must be placed in a memory resident workspace buffer. Since there is a relatively large access delay on a disk (due to seek and latency times), the workspace buffer should be as large as possible to maximize the amount of data transferred in a single disk access.

Coupled with this, however, is the constraint that the total program and buffer size cannot exceed available memory.

In a virtual memory system (e.g., DEC VAX 11/780), the available memory space is limited only by the amount of available secondary storage. The operating system pages information in and out of memory to maintain the portion of the buffer being operated on in memory.

In non-virtual systems, such as the Sigma 5, the available buffer space is limited to the amount of physical memory allocated to the user. If this memory space is too small (i.e., unable to contain the largest file which will be edited), a paging scheme similar to that of the virtual system must be implemented.
Typically, a set number of lines (page) is read into the workspace buffer. The user may then edit lines within the page. When a reference is made to a line beyond the end of the page, the current page is written to the output file and a new page is read from the input file. References to a line before the beginning of the page cause the remainder of the input file to be copied to the output file, and the output file is reopened as the new input file.

The system on which this editor was implemented had a 32K word block of memory available, which is sufficient for the program and a workspace buffer of approximately 1100 lines. Since the great majority of files at this facility were less than 1100 lines long, no paging facility was implemented.

The commands available to the user have a great influence on the design of the workspace buffer. The requirement that insertions and deletions be allowed at arbitrary points within the file, and that movement within the file (forward and backward) be unrestricted, dictates a doubly-linked list structure for the workspace buffer. A stack of available buffer locations should be maintained so that deleted lines may free their buffer space.

**Edit operations**

Any editing task may be specified in terms of one of three basic operations. These are:

1. Insert a new line
2. Delete an existing line
3. Change an existing line

Additionally, commands must be provided to position to the desired line within the workspace buffer.

These commands form a functionally complete set of commands which can be used to perform any editing operation. Convenience of use can be greatly increased, however, with additional commands to do such things as FIND a particular string, MOVE particular lines from one place to another, and so on.

The commands incorporated into the current version of the editor for the Sigma 5 are shown in Table 1.

Currently, the specification of input and output files is performed by control commands sent to the RBM monitor. An effort is underway to allow the
<table>
<thead>
<tr>
<th>Command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>I NSERT</strong></td>
<td>Insert string following current line. If specified, it is inserted following the current line. Otherwise, insert mode is entered. All lines typed are entered following the current line. Insert mode is exited by entering CR as the first character of a line.</td>
</tr>
<tr>
<td><strong>D ELETE</strong></td>
<td>Deletes current and next n-1 lines. If n is not specified, the current line is deleted.</td>
</tr>
<tr>
<td><strong>+</strong></td>
<td>Advances the current position pointer n lines. If n is not specified, 1 is assumed.</td>
</tr>
<tr>
<td><strong>-</strong></td>
<td>Backs up the current position pointer n lines. If n is not specified, 1 is assumed.</td>
</tr>
<tr>
<td><strong>T OPO</strong></td>
<td>Moves current position pointer to top of workspace.</td>
</tr>
<tr>
<td><strong>B OTTO</strong></td>
<td>Moves current position pointer to bottom of workspace.</td>
</tr>
<tr>
<td><strong>P RINT</strong></td>
<td>Prints current and next n-1 lines. If n is not specified, 1 is assumed.</td>
</tr>
<tr>
<td><strong>C HANGE</strong></td>
<td>Changes first occurrence of old string in current line to new string. First non-blank character is taken to be the delimeter. If trailing delimeter is not specified, new string is assumed to extend to the end of the line.</td>
</tr>
<tr>
<td><strong>TAB</strong></td>
<td>Sets up to 11 tab positions. The command has no effect if no tabs are specified.</td>
</tr>
</tbody>
</table>

**NOTE:** Braces indicate an optional field. Reserved words are indicated by capital letters, lowercase letters indicate user-supplied fields.

CR indicates carriage return.

\$ indicates a required blank.
TABLE 2. SAMPLE JOB STREAM

! JOB EDIT, IFILE
! STDLB (SI, D3, IFILE) (1)
! STDLB (SO, D3, OFILE)
! EDIT (2)
! FIN

NOTES:
1. The STDLB command should be used to assign SI to the edit input file and SO to the edit output file.
2. In order for EDIT to be loaded into memory, the user must do an FMEM Ø Keyin.
files to be specified from the edit terminal (in this case a Tektronix 4002). The sample job stream for the current configuration is shown in Table 2.

Also under development are modifications which will allow the editor to run as a foreground program with interrupt driven I/O routines. This will allow background batch processing to run concurrently with edit I/O operations. Since most of the time used by an editor is in waiting for user response and in doing I/O, interrupt driven operation of the editor will have very little effect on the turnaround time for background jobs.

Appendix 1 shows a sample edit session using the current version of the editor.

Appendix 2 is a listing of the editor program.
III. CONCLUSIONS AND RECOMMENDATIONS

The interactive text editor described in this report has been installed on the Sigma 5 system located in the Guidance and Control Analysis facility at Redstone Arsenal.

Use of this processor has resulted in a significant decrease in time required to create and update files of symbolic data.

It is recommended that this processor be expanded to include capabilities such as to FIND a particular string, to MOVE a line or group of lines to some location, and a capability to define MACRO commands consisting of several edit commands.

These enhancements should reduce even further the amount of time required for symbolic data entry and maintenance.
REFERENCES


APPENDIX 1

SAMPLE EDIT SESSION
*BOF*
EDI>
IN >THIS DEMONSTRATES THE INSERT MODE.
IN >WHEN AN INSERT COMMAND IS FOLLOWED
IN >ONLY BY A CARRIAGE RETURN THE INSERT
IN >MODE IS ENTERED. TO EXIT THE INSERT
IN >MODE ENTER A LINE CONSISTING ONLY
IN >OF A CARRIAGE RETURN.
IN >
EDI>TOP
*BOF*
EDI>PRINT 5
*BOF*
>THIS DEMONSTRATES THE INSERT MODE.
>WHEN AN INSERT COMMAND IS FOLLOWED
>ONLY BY A CARRIAGE RETURN THE INSERT
>MODE IS ENTERED. TO EXIT THE INSERT
EDI>
>OF A CARRIAGE RETURN.
EDIT>CHANGE /CARRIAGE/HORSE AND BUGGY/
>OF A HORSE AND BUGGY RETURN.
EDIT>TOP
*BOF*
EDIT>P
*BOF*
>THIS DEMONSTRATES THE INSERT MODE.
>WHEN AN INSERT COMMAND IS FOLLOWED
>ONLY BY A CARRIAGE RETURN THE INSERT
>MODE IS ENTERED. TO EXIT THE INSERT
>MODE ENTER A LINE CONSISTING ONLY
>OF A HORSE AND BUGGY RETURN.
EDIT>-5
>THIS DEMONSTRATES THE INSERT MODE.
EDIT>DELETE
>WHEN AN INSERT COMMAND IS FOLLOWED
EDIT>TOP
*BOF*

EDI>P 3

*BOF*

> WHEN AN INSERT COMMAND IS FOLLOWED
> ONLY BY A CARRIAGE RETURN THE INSERT
> EDI>BOTTOM
> OF A HORSE AND BUGGY RETURN.
> EDI>I THE INSERT COMMAND CAN ALSO BE
> EDI>I USED IN A SINGLE LINE INSERTION MODE
> EDI>I LIKE THIS.
EDI>-5
> MODE IS ENTERED. TO EXIT THE INSERT
EDI>PRINT 5
> MODE IS ENTERED. TO EXIT THE INSERT
> MODE ENTER A LINE CONSISTING ONLY
> OF A HORSE AND BUGGY RETURN.
> THE INSERT COMMAND CAN ALSO BE
> USED IN A SINGLE LINE INSERTION MODE

EDI>
(TO TERMINATE THE EDIT SESSION AND
WRITE THE WORKSPACE BUFFER TO THE OUTPUT
FILE USE THE END COMMAND)

ED1>END
*END*
APPENDIX 2

LISTING OF EDITOR PROGRAM
EDIT --- INTERACTIVE TEXT EDITOR FOR THE SIGMA 5

The program edits a specified input file to a specified output file according to commands issued from the terminal graphics console.

The program runs in foreground which always editing to be performed concurrently with background processing.

**INTEGER** (A-P)

The workspace buffer is a doubly-linked list. Variable head points to the head of the list, tail to the tail of the list. f.L_(1) contains the forward link for the input node and .f.L_(1) the backward link for the input node. node 1 of the list consists of a card image which is saved in different nodes.

Additionally, a stack of free nodes (AVAIL) is maintained so that deletions can free buffer stations for later use.

```
100 C0MHAU32, 32H33, 33HAV4, 43HAV5.
200 C0MHAU, 32H33, 33HAV4, 43HAV5.
300 D3M3N33, 1, 1 NV4, NV4, 1 NV4.
400 C0MHAU52, 32H40, 13HAV5, 54HAV5.
500 D3M3N33, 119, 134, 1344, 1344.
600 C0MHAU70, 1344, 1344, 1344, 1344.
700 C0MHAU72, 1344, 1344.
800 C0MHAU73, 1344, 1344.
900 C0MHAU74, 1344, 1344.
1000 C0MHAU75, 1344, 1344.
1100 C0MHAU76, 1344, 1344.
1200 C0MHAU77, 1344, 1344.
```

1. Initialize line counter
2. Initialize current buffer position
3. Initial prompt
4. Call main
5. Call edit
6. Set up default tabs positions defined by TAB510
7. 80.20.10.4.10.8
8. 40.
9. TAB511.
10. Set IP initial tab position
11. Call TAB511, TAB511, TAB511.
12. Call TAB511, TAB511, TAB511, TAB511, TAB511, TAB511, TAB511, TAB511, TAB511.
13. Set IP input and output files
14. Call input
15. Read page into workspace buffer
16. Call BUZZED, BUZZED, BUZZED.
17. Continue
18. Set command from input and return value
19. Call close
20. Call BUZZED
21. Set command from input and return value
22. Call close
23. Call BUZZED
24. Continue
SUBROUTINE INSET

SUBROUTINE INSET PROCESSES INSET FOR INPUT AND OUTPUT FILES
AND INITIATES VALIDITY CHECKS ON THESE FILES. SUBROUTINE WILL
BE CALLED IF A CORRECT INPUT AND OUTPUT FILE HAS BEEN ENTERED.

CALLED BY: MAIN
CALLS: SUBROUTINE CHECK

**********************************************************************

COMMON / IIN / IN, JIN, TAENA, IFNAM, BAERA, SFNAM

EXPONENT ENDR (4,7)

IFNAM WILL CONTAIN THE INPUT FILE NAME, SFNAM THE OUTPUT FILE NAME
DIMENSIONS IFNAM(2), SFNAM(2)
VARIABLES TAENA AND BAERA WILL CONTAIN THE INFILE AREA
AND THE OUTFILE AREA, RESPECTIVELY.

CONTINUE

PREPARE FOR INPUT FILE
CALL PREPMT ('ENTER INPUT FILE #1')
READ FILE AND VERIFY CORRECT AREA, FORMAT AND ACCESS
CALL CHECK(TAENA, IFNAM)
IF (TAENA /= 0, THEN(1) TO 10
CONTINUE

PREPARE FOR OUTPUT FILE
CALL PREPMT ('ENTER OUTPUT FILE #1')
READ FILE AND VERIFY CORRECT AREA, FORMAT AND ACCESS
CALL CHECK (BAERA, SFNAM)
IF (BAERA /= 0, THEN(1) TO 20
RETURN
END
**SURROGATE SELECT**

**SURROGATE READS COMMAND FROM TERMINAL AND ATTEMPTS TO MATCH FIRST**

**TOKEN WITH TOKENS IN THE COMMAND TABLE.**

**CALLED BY: AMM**

**CALLS: PMODE, INSYNC, eigenen**

**DIMENSION TABLE(11x11)**

**DATA TABLE**

<table>
<thead>
<tr>
<th>INDEX</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>n</td>
<td>r</td>
<td>m</td>
<td>t</td>
<td>r</td>
<td>f</td>
<td>l</td>
<td>a</td>
<td>k</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>e</td>
<td>l</td>
<td>e</td>
<td>t</td>
<td>e</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**DATA VARIABLES**

**CALL**

**CONTINUE**

**INNER LOOP SCANS INNER CHAR AGAINST CHAR FROM VALID COMMAND**

**TABLE WITH INTEGRAL CHARACTERS WHICH EVERY TABLE CHAR IS AGAINST VARIABLE**

**MODE IS INCREMENTED WHENEVER AN ASTERISK IS ENCOUNTERED IN THE**

**TABLE, AND IS INTERPRETED AS END OF**

**CHARS.**

**MACH*E**

**CHARS TABBED FAR ARE NOT SUFFICIENT TO DEFINE THIS CMD**

**MACH**

**CHARS TABBED FAR ARE SUFFICIENT BUT NOT COMPLETE**

**MACH**

**ALL CHAR IN THE COMMAND HAVE BEEN SCANNED**

**3 3 $0 1, VMCG**

**MODE = 0**

**10 20 PMODE 8**

**15 SET CHT FILL TABLES IF = $0 MODE 10 AND SET NEXT**

**CONTINUE**

**x = SETCH2TABLELSMOSMOL$$**

**IF (n$5, x) \$0 (n$5, 10) \$0 (n$5, 10) \$0 (n$5, 10)**

**10**

**SET CHT FILL COMMAND XFER**

**y = SETCH2TABLELSMOSMOL$$**

**IF y IS A CHAR OR A NUMBER AND MODE > 10 WE ARE DONE**

**11**

**10 IF (n$5, x) \$0 (n$5, 10) \$0 (n$5, 10) \$0 (n$5, 10) \$0 (n$5, 10) \$0 (n$5, 10)**

**10**

**RETURN**

**10**

**10 IF MACH*E THIS COMMAND IS 100 ANOTHER CMD NEXT CMD**

**10**

**10 IF MACH*E 103, 20 30 10 91**
CASE 2: IF VHA = 0 WE ARE BEFORE THE HEAD

IF (VBA = 0) GB 19 19 37
F (VBA) = 0
R (VBA) = 0
R (VBA-HEAD) = 1
VHA = 1
VHD = 1
LINE = 1
39 19 45
CONTINUE

CASE 3: WE ARE SOMEWHERE WITHIN THE LIST, POSSIBLY AT THE TAIL

R (VBA) = VHD
F (VBA) = F (VBA-HEAD)
IF (VBA = VBA-HEAD) R (VBA-HEAD) = 1
F (VBA-HEAD) = 1
CONTINUE

IF (VBA = TAIL) TAIL = 1
VHA = 1
LINE = LINE + 1
113 (1532 1373 0) RETURN
CALL INCR (C340006,90,43)
CALL JUMP (.C400006J) C400006J (3) + 4
343 = 0
CONTINUE
CALL PLOT (*350+4)
CALL TABS*(TABS*2) + 2, TABS*3 + 2, TABS*4 + 2
1 1 TABS*5 + 2, TABS*6 + 2, TABS*7 + 2, TABS*8 + 2
2 2 TABS*9 + 2, TABS*11 + 2, TABS*13 + 2
SUBROUTINE DELETE

SUBROUTINE DELETS N NODES FROM CURRENT BUFFER AND
RETURN NODES TO AVAILABLE STACK.

IMPLICIT INTEGER(4,5)

COMMON /C3/ C3(42), C3(J), C3, B3, V3, T3(11)
COMMON B hinter, LINING, HEAD, TAIL, AVAIL, TOP

INTEGER, EST. V3
DIMENSION BUFFER(E120), T4(120), T1(14(120)), T2(14(120)), AVAIL(120)

SET NUMBER OF LINES TO DELETE INTO VM

DECIDE (P1, (P2) = 0, VM)
FORMAT (W)

DELETES VM LINES

IF 200 > VM
IF LIST IS EMPTY (HEAD = TAIL = VM) OR WE ARE AT END (VM = 0)
PRINT ERROR AND RETURN
IF VM > 0 CALL DELETE (VM + 1) RETURN
PLACE THE NODE ABOUT TO BE DELETED ON THE AVAILABLE STACK
TOP = TOP + 1
AVAIL(VM) = VM
IF LIST HAS NOY ONE NODE (HEAD = VM), OR IF WE ARE DELETING
THE TAIL NODE (TAIL = VM), WE WILL PRINT ERROR. IF THERE IS ONLY
ONE NODE, HEAD, TAIL, AND VM WILL BE SET TO 0 AND RETURN.
OTHERWISE TAIL AND VM WILL BE SET TO 0, HEAD, TAIL, AND LINE VM WILL
BE PRINTED.

REPEAT 100, WHILE VM > 0, TAIL
CALL PRINTER (VM-1, VM)
VM = VM
TAIL = VM = 0: INC(VM)
IF (HEAD = 0, T6, HEAD = 0) RETURN
INC(VM) = 1
CALL PRINTER (VM, VM)
RETURN
END CONTINUE

IF WE ARE ABOUT TO DELET
IF WE ARE ABNER TO DELET A FILE THAT IS NOT AT THE TAIL,
WE MAY BE AT THE HEAD. WE WILL SET 0, T4, SET VM, PRINTING,
FROM T1(14(VM)) = 0, T1(14(VM)) = 0, T1(14(VM)) = 0, T1(14(VM)) = 0
VM = T1(14(VM))
END OF NEXT DELETION
END CONTINUE
CALL PRINTER (VM, VM)
END
In the current position i at which the transition z is made, the transition z makes the next state y and moves to the next state y. The transition z makes the next state y and moves to the next state y. The transition z makes the next state y and moves to the next state y.

DIMENSION BUFFER(2000), IR(14)((1200), 3.141(1200), AVAIL(1200))

IF LIST IS EMPTY, PRINT "END" AND RETURN
IF HEAD = 0, CALL JIFSCRD(149770, 3) RETURN

DECIDE (HEAD > BUFFER) IF YES = 1

IF (HEAD = 0) AND (1 + 37 = 11) AHEAD(10) TO 18
REPEAT 16 WHILE VN > 0
CALL JIFSCRD(100000, 3)
34 TO 35
CONTINUE

IF (3 + VN = 11) VN = 214 VN(3)
REPEAT 17 WHILE VN > 0
CALL JIFSCRD(1*399, 3)
34 = 141
CALL PRINT BUFFER(1494)
RETURN
CONTINUE
17 CALL PRINT BUFFER(1494)
CONTINUE
END
<table>
<thead>
<tr>
<th>Recipient</th>
<th>No. of Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>Commander</td>
<td>12</td>
</tr>
<tr>
<td>Defense Documentation Center</td>
<td></td>
</tr>
<tr>
<td>ATTN: DDC-TCA</td>
<td></td>
</tr>
<tr>
<td>Cameron Station</td>
<td></td>
</tr>
<tr>
<td>Alexandria, VA 22314</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td>5</td>
</tr>
<tr>
<td>US Army Research Office</td>
<td></td>
</tr>
<tr>
<td>ATTN: DRXRO-PH, Dr. R. Lontz</td>
<td></td>
</tr>
<tr>
<td>P.O. Box 12211</td>
<td></td>
</tr>
<tr>
<td>Research Triangle Park, NC 27709</td>
<td></td>
</tr>
<tr>
<td>US Army Research &amp; Standardization Group (Europe)</td>
<td>1</td>
</tr>
<tr>
<td>ATTN: DRXSNE-RX, Dr. Alfred K. Kedoluha</td>
<td></td>
</tr>
<tr>
<td>Box 65</td>
<td></td>
</tr>
<tr>
<td>FPO New York 90510</td>
<td></td>
</tr>
<tr>
<td>Commander</td>
<td>1</td>
</tr>
<tr>
<td>US Army Material Development &amp; Readiness Command</td>
<td></td>
</tr>
<tr>
<td>ATTN: Dr. James Bender</td>
<td></td>
</tr>
<tr>
<td>Dr. Gordon Bushy</td>
<td></td>
</tr>
<tr>
<td>5001 Eisenhower Avenue</td>
<td></td>
</tr>
<tr>
<td>Alexandria, VA 22333</td>
<td></td>
</tr>
<tr>
<td>Hq, Department of the Army</td>
<td>1</td>
</tr>
<tr>
<td>Office of the DCS for Research, Development &amp; Acquisition</td>
<td></td>
</tr>
<tr>
<td>ATTN: DAMA-ARZ</td>
<td></td>
</tr>
<tr>
<td>Room 3A474, The Pentagon</td>
<td></td>
</tr>
<tr>
<td>Washington, DC 20310</td>
<td></td>
</tr>
<tr>
<td>ONUSDR&amp;E</td>
<td>1</td>
</tr>
<tr>
<td>ATTN: Mr. Leonard R. Weisberg</td>
<td></td>
</tr>
<tr>
<td>Room 301079, The Pentagon</td>
<td></td>
</tr>
<tr>
<td>Washington, DC 20301</td>
<td></td>
</tr>
<tr>
<td>Director</td>
<td>1</td>
</tr>
<tr>
<td>Defense Advanced Research Projects Agency</td>
<td></td>
</tr>
<tr>
<td>1400 Wilson Boulevard</td>
<td></td>
</tr>
<tr>
<td>Arlington, VA 22209</td>
<td></td>
</tr>
<tr>
<td>ONUSDR&amp;E</td>
<td>1</td>
</tr>
<tr>
<td>ATTN: Dr. G. Ganota</td>
<td></td>
</tr>
<tr>
<td>Deputy Asst. for Research (Research in Advanced Technology)</td>
<td></td>
</tr>
<tr>
<td>Room 301067, The Pentagon</td>
<td></td>
</tr>
<tr>
<td>Washington, DC 20301</td>
<td></td>
</tr>
</tbody>
</table>
## DISTRIBUTION (Concluded)

<table>
<thead>
<tr>
<th>Code</th>
<th>Name</th>
<th>Copies</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRSMI-R</td>
<td>Col. Fiest</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Mr. Todd</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Mr. Plunkett</td>
<td>1</td>
</tr>
<tr>
<td>-RGN</td>
<td>Mr. Murdock</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>Dr. Pastrick</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Dr. Kelly</td>
<td>1</td>
</tr>
<tr>
<td>-RPT</td>
<td>Dr. Hartman</td>
<td>2</td>
</tr>
<tr>
<td>-RR</td>
<td>Dr. Bennett</td>
<td>1</td>
</tr>
<tr>
<td>DRSMI-LP</td>
<td>Mr. Voigt</td>
<td>1</td>
</tr>
</tbody>
</table>

IIT Research Institute
ATTN: GACIAC
10 West 35th Street
Chicago, Illinois 60616

39