NOAA Technical Memorandum ERL APCL-17

U.S. DEPARTMENT OF COMMERCE
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
Environmental Research Laboratories

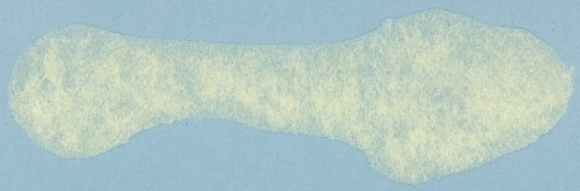
TINY
A Short Program for TTY Interface to a NOVA Minicomputer

UWE HERBERT GROTE

Atmospheric Physics and Chemistry Laboratory
BOULDER, COLORADO
September 1974

ONMENTAL RESEARCH LABORATORIES

HERIC PHYSICS AND CHEMISTRY LABORATORY



IMPORTANT NOTICE

Technical Memoranda are used to insure prompt dissemination of special studies which, though of interest to the scientific community, may not be ready for formal publication. Since these papers may later be published in a modified form to include more recent information or research results, abstracting, citing, or reproducing this paper in the open literature is not encouraged. Contact the author for additional information on the subject matter discussed in this Memorandum.

NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION

BOULDER, COLORADO

NOAA Technical Memorandum ERL APCL-17

TINY,
A Short Program for TTY Interface to a NOVA MINICOMPUTER

Uwe Herbert Grote

Atmospheric Physics and Chemistry Laboratory
Boulder, Colorado
September 1974

UNITED STATES
DEPARTMENT OF COMMERCE
Frederick B. Dent, Secretary

NATIONAL OCEANIC AND
ATMOSPHERIC ADMINISTRATION
Robert M. White, Administrator

Environmental Research
Laboratories
Wilmot N. Hess, Director

DISCLAIMER

The Environmental Research Laboratories do not approve, recommend, or endorse any proprietary product or proprietary material mentioned in this publication. No reference shall be made to the Environmental Research Laboratories or to this publication furnished by the Environmental Research Laboratories in any advertising or sales promotion which would indicate or imply that the Environmental Research Laboratories approve, recommend, or endorse any proprietary product or proprietary material mentioned herein, or which has as its purpose an intent to cause directly or indirectly the advertised product to be used or purchased because of this Environmental Research Laboratories publication.

TABLE OF CONTENTS

TINY

A Short Program for TTY Interface to a NOVA Minicomputer

Uwe Herbert Grote[1]

1. USER MANUAL

1.1 General

This program makes possible direct communication in machine code between a teletypewriter (TTY) and a NOVA minicomputer. It can debug or input short programs; it has a dumping capability and a simple editing feature which compensate for limited memory. It uses octal numbers. In the appendices are basic NOVA instructions, examples of locations, the program listing, flow charts for certain operations, and a table showing sequence of events.

1.2 Controls

SPACE	Print present (or next) location and its contents
CARRIAGE RETURN	Print previous location and its contents
LINE FEED	Delete last word
RUBOUT	Delete last character
ESCAPE	Return to start
L	Insert last word as low limit
H	Insert last word as high limit
P	Print contents of locations between low and high limit
Z	Enter six zeroes as contents of location

---

[1]Present address: Naval Air Development Center, AETD, Warminster, Pa.

1.3 Operation

Depressing a digit key (0 to 7) causes the digit to be printed and to be incorporated into a word that is stored in memory. Depressing a key that represents neither a valid digit nor a control function will not contribute to the word; a question mark will be printed.

Depressing the "ESC" key brings control back to the starting location. A carriage-return and line-feed are initiated and a dot is printed to indicate that the next five digits will be entered as an address.

After five digits have been entered, a space is printed, and the five-digit word is stored as an address. The next six digits represent the contents entered into the defined address.

After the sixth digit of the contents has been entered, a carriage return and line feed occur. The computer is now ready to take the next six digits and store them in the next location. Since there are no control characters for this operation, it is necessary to print all digits (five for an address and six for the contents of a memory location) including leading zeroes.

This procedure provides easy entry to a program in machine language. The first five digits represent the starting address, and each set of six digits thereafter is entered as the contents of the next consecutively numbered location.

Example:

.04652 514327
641723
000413

The dot is printed automatically, indicating the start of an address. The space, carriage return, and line feed occur automatically; only the numbers are inserted at the keyboard.

04652 is the starting address; the addresses and their new contents are

04652 514327
04653 641723
04654 000413

1.4 Corrections

If the operator types a wrong digit, he depresses the RUBOUT key. This prints a left-directed arrow and brings control back to the previous digit. This can be repeated several times. Each time, control goes back one more character.

Several special cases have to be considered:

a. The last character entered is part of the five digit location word. If the RUBOUT key is depressed so many times that control would go beyond the address, a series of operations duplicates those produced by pressing the "ESC" key: Carriage-return and line-feed are initiated, and a dot is printed. The first number entered after the last RUBOUT is then again the first number of an address.

b. The last character entered is the fifth character of the address. A space has been printed to indicate that the address is done. (No new number has been entered.) If the RUBOUT key is depressed now, control goes back to the last digit of the address.

c. The last character entered is part of the contents word. If the RUBOUT key is depressed so many times that control would go beyond the first digit of the contents word, it stops automatically, so that regardless of how many times the RUBOUT key is pressed, control stays at the first digit of the contents word. The first number entered is then the first digit of the contents word.

d. The last character entered is the last digit of the contents word. Carriage-return and line-feed have occurred, and the program is ready to accept the next word. If the RUBOUT key is depressed now, control goes back to the last digit of the contents word just finished.

Example:

.027←516 ←5

The word is 02515. The dot indicates it is an address. The keys depressed were: 027 RUBOUT 516 RUBOUT 5

Example:

.05←←←

.

No word has been stored. The keys depressed were: 05 RUBOUT RUBOUT RUBOUT

Example:

13←←←053276

←2

The word is 053272. The keys depressed were: 13 RUBOUT RUBOUT RUBOUT 05327 RUBOUT 2

If the operator wants to delete the last word, he depresses the LINE FEED key. This causes a right open inequality sign to be printed. In case this happens in the beginning while the starting address is being entered (as long as not more than five keys have been entered), control goes back to start. If the operator depresses the LINE FEED key while a word for storage is being composed, control goes back to the beginning of this word. If the operator depresses the LINE FEED key after a contents word has been finished, the new word will be composed again at the same location. If the operator depresses the LINE FEED key several times while entering an address, a < sign is printed each time. Carriage return and line feed are initiated as if the "ESC" key had been depressed. If the operator depresses the LINE FEED key several times in succession while entering a contents word, a < sign is printed and control goes back one word each time the key is depressed.

Example:

127653

.01157 <

.06721

The word 06721 is entered as the new starting address. The keys depressed after 127653 were: ESC 01157 LINE FEED 06721

Example:

052<154716
043726
<172135

The words in storage are 154716 and 172135. The keys depressed were:
052 LINE FEED 154716043726 LINE FEED 172135

Example:

031276
152341
03<<<024513

The word 024513 replaces the word 031276.

1.5 Adjacent Locations

Pressing the space-key causes a printout of a location and its contents. If an address has been entered, this address and its contents will be printed. If an address and its contents have been printed, pressing the space-key causes the next location and its contents to be printed.

Pressing the carriage return key causes the last location and its contents to be printed.

Example:

.06325
06325 000000

[The content of 06325 is assumed to be zero.] The keys depressed were:
06325 SPACE

Example:

127364
05632 127364
05633 721652

[The content of 05638 is assumed to be 721652.] The keys depressed were: 127364 SPACE SPACE

Example:

05427 003236
05426 274106

It is assumed that 05427 003236 has been printed. Depressing the carriage return prints the preceding address and its contents.

1.6 Limits

After a location has been entered, it can be stored as an upper or lower limit for a printout, as explained in section 1.7. In this process, the last digit is set equal to zero so that complete lines will be printed. Pressing the L or H button causes the printing of the letter L or H, carriage return, line feed, and the printing of a dot.

Example:

.05002 L
.05126 H

The lower limit is 05000; the upper limit is 05127.

1.7 Dump

Depressing the P-button causes the contents of the memory to be printed. The lower limit is the address stored as lower limit, and the upper limit is the address stored as high limit plus seven. If upper and lower limit are the same, the contents of the seven consecutive locations starting with the low limit will be printed.

1.8 Zero

Depressing the Z-button enters six zeroes as the contents of the location. It eliminates having to depress the "0" key six times. Similarly, five zeroes can be inserted by depressing the "Z" and "RUBOUT" buttons. The Z-button cannot be used to "zero" the five-digit address word.

Example:

```
.05001 Z
137652
Z
←2
```

The addresses and their contents are as follows:

```
05001   000000
05002   137652
05003   000002
```

2. SUBROUTINES

This section describes significant subroutines of program TINY. Routines not described are straightforward and require no further explanation. The routines described are: STORE, LIST, OPRTN, ESP, RUB/LINE, and PRW.

2.1 Routine STORE

Routine STORE is used to enter a word into a certain location. The digits accepted by the READ routine must be organized to a word. A counter NUM determines how many places(three octal bits each) the digit must be shifted left to be located at the proper place within the word. This number is negative for the countdown. Since an address word has only five digits, the first input digit is shifted four places (NUM = -4); the first input digit for a stored word is shifted five places (NUM = 5).

From the READ routine, AC 0 contains the accepted digit in ASCII notation and AC 1 contains a 60 (octal), the basis of ASCII numbers. To verify which key was depressed, the selected digit is printed by subroutine PUTC. A flag (FLAG) is set at this time for a later routine (ESP). AC 1 is then subtracted from AC 0 to deliver the pure digit. Loading the counter NUM next, and checking it for zero, will show whether the accepted digit is the last in the word. If it is, no

shifting is necessary, so control jumps directly to EWO. Otherwise, a shift loop (SHIFT) is entered. NUM is not changed in the shift loop; only AC 1, into which NUM was loaded, is counted down. Since the word was originally zero, and since only one digit is present, the properly shifted digit can be added to the word, which is then stored.

Next, loading NUM into AC 1 and placing the incremented value into AC 2 stores the incremented value in NUM, while also testing the old value of NUM by a move instruction. NUM becomes zero when the last digit is read in. If this happens, the word needs to be stored; if not, more digits have to be read.

Once a word is fully assembled it must be decided whether it should be stored as an address (in ADDR) or as the contents of this address. The flag for deciding is AOW. It is originally -1. Only when a word is initially read in should it be stored in ADDR. Incrementing AOW provides zero in that case. For all positive values of AOW, control goes to MEM. The indication to the keyboard operator that an address is stored is that a space (C40) is printed.

The following part of the routine, starting at CONS and continuing to MEM, is also used by other parts of the program. It initializes the word with zeroes and stores -5 in the counter NUM before control returns to READ. A flag (AONU) is set for a later routine.

After the initial word, a new word is stored by the routine starting at MEM. AC 0 contained the word that is now stored at the location indicated by ADDR, and which is also saved in HOLD for later editing routines. The keyboard operator is notified by a carriage return and line feed that the word has been stored. The address is then incremented and the program is ready to receive the next character from the READ routine.

2.2 Routine LIST

The contents of locations specified by the upper and lower limits are dumped with this routine. A subroutine PRW (print word) converts the octal instruction into ASCII form for printout. For a dump, nine

words are printed per line, the first word being the starting address for a set of eight contents words to follow. PRW will also print one or two words per line as called for in routines ESP and ECR.

The address of NLINE is stored in JOUT. NLINE is the beginning of instructions for each new line to be printed. The lower limit is initially set as the current address CADR. Next the upper limit is loaded into AC 2. The current address is compared with it to insure that the upper limit is not exceeded. If the upper limit has been reached, program control goes back to START. Carriage return and line feed are initiated to start each new line. The number 11 (octal) is stored in COUNT for use in PRW to obtain the nine words per line. Minus one (-1) is loaded into NSP (number of spaces) of PRW to provide for one space printed behind the address and one space printed in front of the contents.

2.3 Routine PRW

This routine is entered whenever a word and/or address is to be printed. AOW is initialized to -1 to identify the beginning of an address. To print the address, the instruction in ADIN is transferred to EXTR. This instruction can be incremented to load the contents of successive locations starting with TTH.

First, the current address, i.e., the address to be printed, is loaded into AC 1. AC 2 is then loaded with 10000 (octal), and AC 0 is loaded with 60 (octal), the basis of the ASCII numbers. AC 2 can now be successively subtracted from AC 1 until the result is negative. Each time the subtraction results in a positive value, AC 0 is incremented. At the conclusion of the subtractions, AC 0 is incremented. At the conclusion of the subtractions, AC 0 will contain the ASCII code corresponding to the first digit of the word. An "add" instruction restores the last positive value of AC 1. The ASCII character in AC 0 is printed by subroutine PUTC.

This process can also extract the next digits. EXTR is incremented to load the other octal multiples in order. AC 2 is checked each time

to see if the last digit loaded into AC 2 was 1 (one). As long as AC 2 does not contain 1, control goes back to EXTR and extraction of the next ASCII character. AOW is incremented to reflect address completion. After an address has been printed, CADR is decremented to compensate for an incrementation a few steps later in the routine. The incrementation is actually part of the routine starting at NWO.

To print the contents of an address, the instruction WOIN is loaded into AC 2 and stored in EXTR. Subroutine SPP prints the desired number of spaces between words. The current address is incremented to prepare for the next word to be printed (a "jump" step is inserted in case the address should be zero). The value of COUNT is decremented. If COUNT is zero, all words have been printed and the routine ends; otherwise AC 1 is loaded with the contents of the next address, and the routine jumps back to EXTR.

2.4 Routine OPRTN

OPRTN is entered when the READ routine accepts a character that does not decode to an octal number. There are nine operations (TOTN = 11).

The octal value 11 is stored in CNTO where it can be counted down. The address of the last instruction in OPRTN is loaded into AC 2 and stored in COMPA. The content of the address in COMPA is an ASCII code. By loading the contents of COMPA into AC 2 and subtracting the ASCII code in AC0 (entered by READ), we can determine if the two characters agree. Meanwhile, COMPA is decremented, so that the address in COMPA contains a jump instruction to the designated operation. If the two ASCII codes agree, the program is directed to that operation. Otherwise, COMPA and CNTO are decremented, and the cycle is repeated by a jump to NEXT. When CNTO reaches zero and the operation is not found, a question mark is loaded into AC0 and printed by subroutine PUTC. The program is then ready to read the next character.

2.5 Routine ESP

This routine is entered when the keyboard operator depresses the space bar. The routine decides whether to print the present or the subsequent address and its contents. The flags used for this are AOW and FLAG.

The first step is to check whether an address has been stored (AOW > -1). If an address has not been stored, control goes back to START. If, however, the keyboard operator has just completed the starting address (AOW = 0), FLAG will be incremented, a carriage return and line feed will be initiated, and the program will enter the current address into CADR. Two words (address and contents) are to be printed. NSP is given the value -1 to specify one space. COUNT is set to 2 for PRW, indicating a total of two words to be printed. The return address from PRW is stored in JOUT. The ESP routine is exited with a jump to RET (STORE).

If the operator has entered the starting address and perhaps two or more contents words, AOW will be greater than zero, and the program should print the last address and its contents. FLAG is incremented and checked to see if it is zero. Since FLAG is set to -1 at the beginning of the STORE routine, FLAG equal to zero after this incrementation indicates that ESP was entered from STORE. The address needs to be decremented only if ESP was entered from STORE. Except for address selection, the ESP routine starting at PP is the same as mentioned in the preceding paragraph.

2.6 Routine RUB/LINE

If the operator makes mistakes, he can erase the last character (RUB) or the entire word (LINE). There are three flags that direct the flow of events through the routine: AOW (address or word), NUM (number of digits left in word), and AONU (old or new word). AONU is required, since the same condition of AOW and NUM can be obtained after a word (or

address) has been completed, or the immediately following word has been erased. AONU is incremented in the RUB/LINE routine and set to -1 in the STORE routine.

Both keys, RUB and LINE, lead to the same procedure. The entry points are different to allow printing an "arrow" for RUB and a "less than" symbol for LINE. AC 0 will retain the ASCII code, either the "arrow" or the "less than" symbol, throughout the routine to direct the proper course of action.

If the address has not been completed (AOW = -1), and all the characters of the partially assembled address are to be erased, depressing LINE will exit the STORE routine and return control back to START. If only one character is to be erased, NUM is checked for -4. NUM = -4 indicates no character has been entered and control goes back to START. A number greater than -4 means that at least one character has been entered and the erase procedure is started (AA): AONU is incremented (to keep record that an erase has been performed), NUM is decremented to its preceding value, and the address/contents are loaded into AC 2. A mask is entered into AC 0 and is shifted so that, when "anded" with AC 2, the last entered character of the address/word is zeroed. The routine is then ready to accept the corrected character from the READ routine.

If the address and part of its contents have been entered (AOW = 0, NUM > -5), and the last character is to be erased, the erase procedure is the same as above. To erase all characters in the contents, control is returned to CONS (STORE).

AOW > -1 and NUM = -5 occur before the first character of each word that is to be entered for the given address. If the address at which the word is to be stored was just entered by the operator, AOW = 0, but if it was incremented by the program, AOW will be positive.

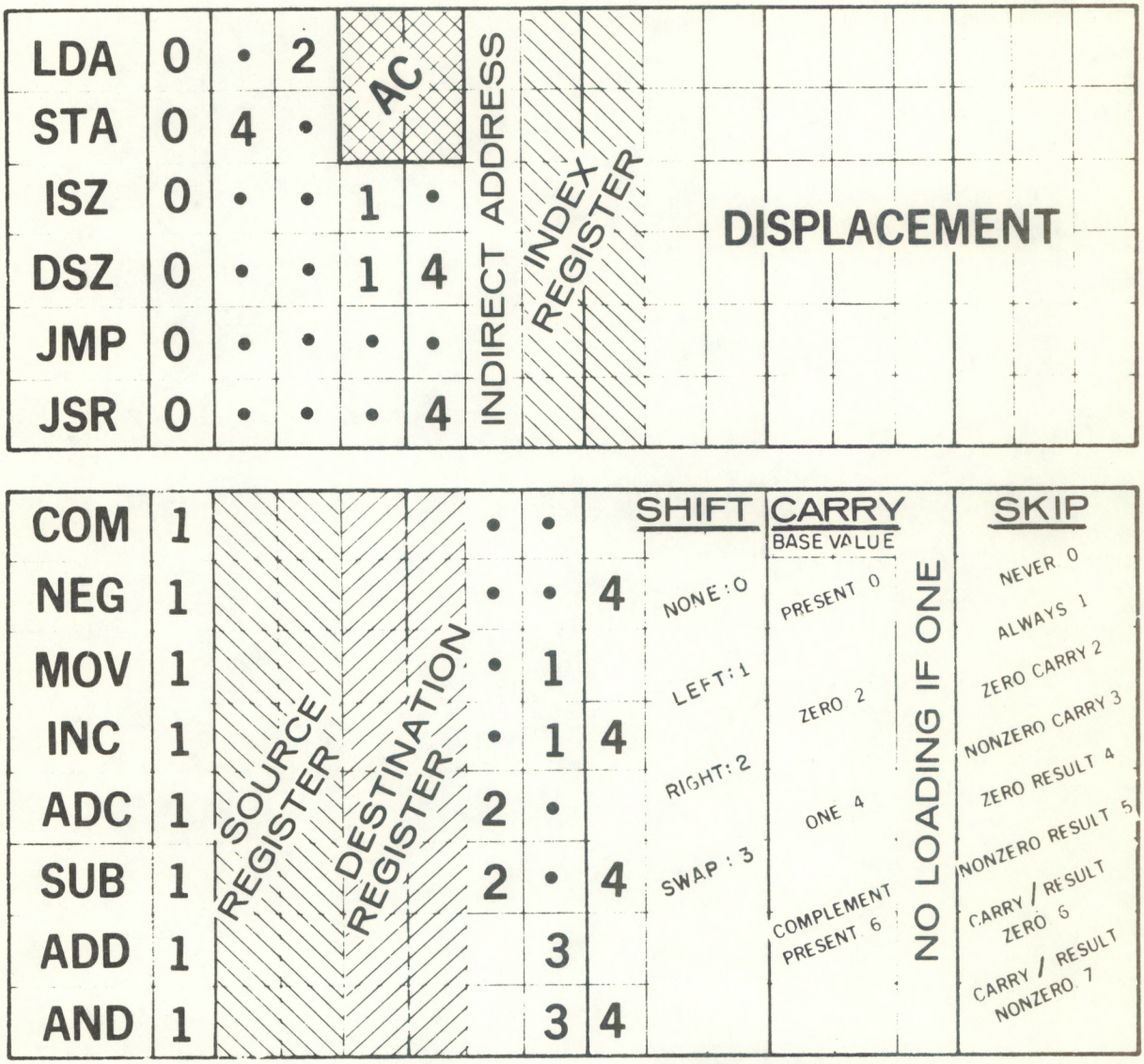
Where AOW = 0, NUM = 5, and the LINE key is depressed, the just completed address will be erased. Control of the program goes to START. There is another time when AOW = 0 and NUM = -5: When a word is erased AOW is also decremented, so that AOW could also be zero and NUM = -5

after a word has been erased. Since further erasure of word or character should not destroy the address in this case, AONU is incremented during the erase routine. Thus with AONU previously incremented, any further request to erase simply returns control to READ. Should AONU still be equal to -1, no erase has been performed; an instruction erases the last character of the address. This is accomplished by setting AOW = - 1, NUM = 0, and taking the address out of HOLD and putting it in AC 2. Erasing again is accomplished by proper shifting of the mask.

Consider now the other case mentioned above, where the address was incremented by the program, so that NUM = -5 and AOW is positive. If AONU was previously incremented, control goes back to READ; otherwise the address is decremented. Once the address has been decremented, the decision is made to erase the last character or the entire word. The word is erased (AC 0 = 74) by jumping to program CONS; the last character is erased by decrementing AOW, setting NUM = 0, and retrieving the last word from HOLD and loading it into AC 2. Erasing is accomplished by proper shifting of the mask.

APPENDIX A

Basic NOVA Instructions



Skips if (AC1)=0	MOV #1,1,SZR MOV # 1,1,SZR
Skips if AC2)≠(AC3)	SUB # 2,3,SNR
Skips if (AC0) < (AC1)	SUBZ # 1,0,SZC
Skips if (AC0) ≤ (AC1)	ADCZ # 1,0,SZC

APPENDIX B

Example of Locations for Program TINY

```
.00400 L
.01060 H
.
00400    006565  126440  044554  126000  044564  024542  044547  020554
00410    006554  060110  063610  000777  060610  024531  123400  024526
00420    107400  030527  146414  000546  006540  176000  054526  122400
00430    024525  125005  000405  103120  101120  125404  000775  024517
00440    123000  040515  040524  024512  131400  050510  125004  000742
00450    010520  000413  040505  020537  006510  126440  044500  024473
00460    044475  126000  044471  000726  042473  032472  050500  006476
00470    010467  000764  024476  125102  000510  152400  052461  010471
00500    006464  126000  044452  000762  024464  125112  000672  014453
00510    125015  000551  014445  000550  006450  030442  024451  133400
00520    107400  122404  000403  050507  000402  050504  000652  024515
00530    044511  024500  044502  030477  024500  132433  000642  006426
00540    030473  050474  152120  050475  002472  000170  000177  177774
00550    000060  000465  177773  177777  177777  177774  000000  001060
00560    000006  000630  000621  000056  001046  001053  001060  177770
00570    000000  030442  050766  030766  050766  032765  014764  112415
00600    002762  014761  014756  000772  020433  006757  000603  000675
00610    000015  000436  000040  002433  000033  000677  000110  000675
00620    000114  000706  000120  000454  000012  000450  000177  000643
00630    000132  000400  001060  000011  000634  000003  000762  000077
00640    177776  000533  000137  000074  000533  000411  000400  024721
00650    125132  002775  125005  000405  010700  000406  014701  000404
00660    010674  000401  006703  030674  050750  152000  050752  150520
00670    050745  030660  050747  004467  002655  020745  000402  020744
00700    004546  030654  024666  125113  000407  101213  002740  034640
00710    156405  002735  000430  034637  156404  000423  125004  000413
00720    101213  002725  010631  002722  014644  000401  126420  044626
00730    030636  000415  010621  002712  014623  101213  002475  000765
00740    101213  002472  010611  000401  014611  030611  020621  024606
00750    125005  000406  101120  101120  101120  125404  000774  113400
00760    052454  002664  152000  052452  030433  050402  024646  030432
00770    020442  146443  101401  147001  000775  004451  010771  151203
01000    000767  012434  000403  014631  004432  030413  050761  004427
01010    010624  000401  014623  000402  002625  026617  000751  030433
01020    033432  100000  010000  001000  000100  000010  000001  000012
01030    000015  000040  000060  000455  000556  000570  030602  020772
01040    063511  000777  061111  151404  000773  001400  063511  000777
01050    061111  101004  001400  171000  020754  004771  020751  004767
01060    001000  000000  000000  000000  000000  000000  000000  000000
.
```

APPENDIX C.

Printout of Program TINY

```
 17155   006565 +    START   JSR     @XCRLF      CARR. RETURN/LINE FEED
 17156   126440              SUB     1,1         CREATE 0
 17157   044554 +            STA     1,WORD      SET WORD = ZERO
 17160   126000      CONT    ADC     1,1         CREATE  -1
 17161   044564 +            STA     1,AOW       SET AOW = -1
 17162   024542 +            LDA     1,CM4
 17163   044547 +            STA     1,NUM       SET NUM = -4
 17164   020554 +            LDA     0,DOT
 17165   006554 +            JSR     @XPUTC      PRINT DOT (.)
-17166   060110      READ    NIOS    TTI         START TTI
-17167   063610              SKPDN   TTI
 17170   000777 -            JMP     .-1         WAIT UNTIL DONE
-17171   060610              DIAC    0,TTI       READ CHARACTER
 17172   024531 +            LDA     1,C177
 17173   123400              AND     1,0         REMOVE PARITY
 17174   024526 +            LDA     1,C170
 17175   107400              AND     0,1         ZERO LAST DIGIT
 17176   030527 +            LDA     2,C60
 17177   146414              SUB#    2,1,SZR     IS CHARACTER A NUMBER
 17200   000546 +            JMP     OPRTN       NO - SELECT OPERATION
 17201   006540 +    STORE   JSR     @XPUTC      YES - PRINT NUMBER
 17202   176000              ADC     3,3
 17203   054526 +            STA     3,FLAG      SET FLAG = -1
 17204   122400              SUB     1,0         SUBTRACT 60 FROM ASCII
 17205   024525 +            LDA     1,NUM       AC1 = NUM
 17206   125005              MOV     1,1,SNR     LAST CHARACTER OF WORD
 17207   000405 +            JMP     FWO         YES - DO NOT SHIFT
 17210   103120      SHIFT   ADDZL   0,0         NO - SHIFT CHARACTER
 17211   101120              MOVZL   0,0
 17212   125404              INC     1,1,SZR     INCR. AC1     IS AC1 = ZERO
 17213   000775 -            JMP     .-3         NO - SHIFT AGAIN
 17214   024517 +    FWO     LDA     1,WORD      YES - LOAD STORED WORD
 17215   123000              ADD     1,0         ADD CHARACTER TO WORD
 17216   040515 +            STA     0,WORD
 17217   040524 +            STA     0,HOLD      SAVE WORD
 17220   024512 +            LDA     1,NUM       SET AC1 = NUM
```

```
17221  131400          INC     1,2
17222  050510 +        STA     2,NUM       INCREMENT NUM
17223  125004          MOV     1,1,SZR     IS AC1 = ZERO
17224  000742 -        JMP     READ        NO - READ NEXT CHARACTER
17225  010520 +        ISZ     AOW         YES INCR. AOW, IS AOW =ZERO
17226  000413 +        JMP     MEM         NO - STORE CONTENTS
17227  040505 +        STA     0,ADDR      YES - STORE ADDRESS
17230  020537 +        LDA     0,C40
17231  006510 +        JSR     @XPUTC      PRINT SPACE
17232  126440    CONS  SUBO    1,1
17233  044500 +        STA     1,WORD      STORE ZERO IN WORD
17234  024473 +        LDA     1,CM5
17235  044475 +        STA     1,NUM       SET NUM = -5
17236  126000          ADC     1,1
17237  044471 +        STA     1,AONU      SET AONU = -1
17240  000726 -        JMP     READ        READ NEXT CHARACTER
17241  042473 +  MEM   STA     0,@ADDR     STORE CONTENTS IN LOCATION
17242  032472 +  RET   LDA     2,@ADDR
17243  050500 +        STA     2,HOLD      SAVE CONTENTS
17244  006476 +        JSR     @XCRLF      CARR.RETURN / LINE FEED
17245  010467 +        ISZ     ADDR        INCREMENT ADDRESS
17246  000764 -        JMP     CONS        FORM NEXT WORD
17247  024476 +  ZERO  LDA     1,AOW
17250  125102          MOVL    1,1,SZC     IS AOW NEGATIVE
17251  000510 +        JMP     WHAT        YES - PRINT QUESTION MARK
17252  152400          SUBO    2,2         NO - CREATE ZERO
17253  052461 +        STA     2,@ADDR     STORE 0 IN CONTENTS
17254  010471 +        ISZ     AOW         INCREMENT AOW,IS AOW=ZERO
17255  006464 +        JSR     @XPUTC      NO- PRINT Z
17256  126000          ADC     1,1         YES- CREATE -1
17257  044452 +        STA     1,FLAG      SET FLAG = -1
17260  000762 -        JMP     RET         READY FOR NEXT WORD
17261  024464 +  ECR   LDA     1,AOW
17262  125112          MOVL#   1,1,SZC     IS AOW NEGATIVE
17263  000672 -        JMP     START       YES-GO TO START
17264  014450 +        DSZ     ADDR        NO-DECREMENT ADDRESS
17265  125015          MOV     1,1,SNR     IS AOW = ZERO
```

```
17266  000551 +          JMP     PRN         YES-PRINT ADDR.AND CONT.
17267  014445 +          DSZ     ADDR        NO-DECREMENT ADDRESS
17270  000550 +          JMP     PP          PRINT ADDR.AND CONT.
17271  006450 +  LIM     JSR     @XPUTC      PRINT L OR H
17272  030442 +          LDA     2,ADDR
17273  024451 +          LDA     1,MSK
17274  133400            AND     1,2         ZERO LAST DIGIT OF ADDRESS
17275  107400            AND     0,1         CREATE 110
17276  122404            SUB     1,0,SZR     SUBTRACT 110 FROM ASCII
17277  000403            JMP     .+3
17300  050507 +          STA     2,LIMHI     RESULT 0,ADDR IS HI LIMIT
17301  000402            JMP     .+2
17302  050504 +          STA     2,LIMLO     RESLT NOT 0,ADDR IS LO LIM
17303  000652 -          JMP     START
17304  024515 +  LIST    LDA     1,XNLINE
17305  044511 +          STA     1,JOUT      STORE ADDRESS OF NLINE
17306  024500 +          LDA     1,LIMLO
17307  044502 +          STA     1,CADR      SET STARTING ADDR=LO LIM
17310  030477 +  NLINE   LDA     2,LIMHI     AC2= HIGH LIMIT
17311  024500 +          LDA     1,CADR      AC1= CURRENT ADDRESS
17312  132433            SUB7#   1,2,SNC     IS CUR.ADDR GREATER HI LIM
17313  000642 -          JMP     START       YES- GO TO START
17314  006426 +          JSR     @XCRLF      NO- CARR.RETURN/LINE FEED
17315  030473 +          LDA     2,TOTN
17316  050474 +          STA     2,COUNT     SET COUNT = 11
17317  152120            ADC7L   2,2
17320  050475 +          STA     2,NSP       SET NSP = -1
17321  002472 +          JMP     @XPRW
17322  000170    C170                      NEEDED TO ZERO LAST DIGIT
17323  000177    C177                      NEEDED TO REMOVE PARITY
17324  177774    CM4                       -4,IDENTIFY LENGTH OF ADDR
17325  000060    C60                       NEEDED TO BREAK DOWN ASCII
17326  017242    XRET            A(RET)    ADDRESS OF RET
17327  177773    CM5                       -5.IDENTIFY LENGTH OF WORD
17330  000000    AONU                      SET WHEN CONTENTS IS BEGUN
17331  000000    FLAG                      SET WHEN A WORD IS BEGUN
```

```
17332  000000        NUM                              INDICATES END OF WORD
17333  000000        WORD                             STORES WORD
17334  000000        ADDR                             STORES ADDRESS
17335  000000        CNTO                             NUMBER OF OPERATIONS LEFT
17336  017405  +     XLAD              A(LAD)         ADDRESS OF LAD
17337  000000        COMPA                            CONTAINS LETTER OR INSTR.
17340  000056        DOT                              ASCII (DOT)
17341  017623  +     XPUTC             A(PUTC)        ADDRESS OF PUTC
17342  017630  +     XCRLF             A(CRLF)        ADDRESS OF CRLF
17343  000000        HOLD                             NEEDED TO SAVE WORD
17344  177770        MSK                              NEEDED TO ZERO LAST DIGIT
17345  000000        ACW                              IDENTIFIES ADDR.OR CONTS.
17346  030442  -     OPRTN   LDA       2,TOTN
17347  050766  -             STA       2,CNTO         SET CNTO = 11
17350  030766  -             LDA       2,XLAD
17351  050766  -             STA       2,COMPA        COMPA = ADDRESS OF LAD
17352  032765  -     NEXT    LDA       2,@COMPA       LOAD CONTENTS OF COMPA
17353  014764  -             DSZ       COMPA          DECREMENT COMPA
17354  112415                SUB#      0,2,SNR        DO ASCII CODES AGREE
17355  002762  -             JMP       @COMPA         YES-GO TO SPECIFIED OPER.
17356  014761  -             DSZ       COMPA          NO- DECREMENT COMPA
17357  014756  -             DSZ       CNTO           DECREMENT CNTO,IS CNTO=0
17360  000772  -             JMP       NEXT           NO- NEXT OPERATION
17361  020433  -     WHAT    LDA       0,QM           YES- OPER. NOT FOUND
17362  006757  -             JSR       @XPUTC         PRINT QUESTION MARK
17363  000603  -             JMP       READ
17364  000675  +             JMP       ECR
17365  000015                                         ASCII(CARR.RETURN)
17366  000436  +             JMP       ESP
17367  000040        C40                              ASCII(SPACE)
17370  002433  +             JMP       @XSTART
17371  000033                                         ASCII(ESCAPE)
17372  000677  -             JMP       LIM
17373  000110                                         ASCII(L)
17374  000675  -             JMP       LIM
17375  000114                                         ASCII(H)
```

```
17376  000706 -                JMP     LIST
17377  000120                                   ASCII(P)
17400  000454 +                JMP     LINE
17401  000012                                   ASCII(LINE FEED)
17402  000450 +                JMP     RUB
17403  000177                                   ASCII(RUBOUT)
17404  000643 -                JMP     ZERO
17405  000132        LAD                        ASCII(Z)
17406  000000        LIMLO                      STORES LOW LIMIT
17407  000000        LIMHI                      STORES HIGH LIMIT
17410  000011        TOTN                       USED IN OPERTN. AND LIST
17411  000000        CADR                       CURRENT ADDRESS
17412  000000        COUNT                      NUMBER OF TIMES THRU PRW
17413  017537 -      XPRW             A(PRW)    ADDRESS OF PRW
17414  000077        QM                         ASCII(QUESTION MARK)
17415  000000        NSP                        NUMBER OF SPACES
17416  000000        JOUT                       JUMP TO RET OR NLINE
17417  000137        ARW                        ASCII(ARROW)
17420  000074        LESS                       ASCII(LESS THAN)
17421  017310 -      XNLINE           A(NLINE)  ADDRESS OF NLINE
17422  017166 -      XREAD            A(READ)   ADDRESS OF READ
17423  017155 -      XSTART           A(START)  ADDRESS OF START
17424  024721 -      ESP     LDA      1,AOW
17425  125132                MOVZL    1,1,SZC   IS AOW NEGATIVE
17426  002775 -              JMP      @XSTART   YES-ADDRESS NOT COMPLETE
17427  125005                MOV      1,1,SNR   NO-IS AOW=ZERO
17430  000405                JMP      .+5       YES-ADDRESS FINISHED
17431  010700 +              ISZ      FLAG      NO-INCR.FLAG,IS FLAG=ZERO
17432  000406 +              JMP      PP        NO-ADDRESS WAS NOT INCR.
17433  014701 +              DSZ      ADDR      YES-DECREMENT ADDRESS
17434  000404 +              JMP      PP
17435  010674 +              ISZ      FLAG      INCREMENT FLAG
17436  000401 +              JMP      .+1
17437  006703 -      PRN     JSR      @XCRLF    CARR.RETURN/LINE FEED
17440  030674 -      PP      LDA      2,ADDR
17441  050750 -              STA      2,CADR    SET ADDR.=CURRENT ADDR.
```

```
17442  152000          ADC     2,2         CREATE -1
17443  050752 -        STA     2,NSP       SET NSP = -1
17444  150520          NEGOL   2,2         CREATE 2
17445  050745 -        STA     2,COUNT     SET COUNT = 2
17446  030660 -        LDA     2,XRET
17447  050747 -        STA     2,JOUT      JOUT=ADDRESS OF RET
17450  004467 +        JSR     PRW         PRINT WORD
17451  002655 -        JMP     @XRET       READY FOR NEXT WORD
17452  020745 -  RUB   LDA     0,ARW
17453  000402 +        JMP     .+2
17454  020744 -  LINE  LDA     0,LESS
17455  004546 +        JSR     PUTC        PRINT ARROW OR LESS SYMBOL
17456  030654 -        LDA     2,NUM       AC2 = NUM
17457  024666 -        LDA     1,AOW
17460  125113          MOVL#   1,1,SNC     IS AOW NEGATIVE
17461  000407 +        JMP     LEFT        NO-ADDRESS IS COMPLETE
17462  101213    RIGHT MOVR#   0,0,SNC     YES-RUB OUT OR LINE FEED
17463  002740 -        JMP     @XSTART     LINE FEED-START OVER
17464  034640 -        LDA     3,CM4       RUB OUT-CONTINUE
17465  156405          SUB     2,3,SNR     HAS A CHAR. BEEN ENTERED
17466  002735 -        JMP     @XSTART     NO- START OVER
17467  000430 +        JMP     AA          YES- ERASE LATEST CHAR.
17470  034637 -  LEFT  LDA     3,CM5
17471  156404          SUBO    2,3,SZR     IS AC2 = -5
17472  000423 +        JMP     A           NO-MODIFY CONTENTS
17473  125004          MOV     1,1,SZR     YES-IS AOW=ZERO
17474  000413 +        JMP     C           NO-CHECK FURTHER
17475  101213          MOVR#   0,0,SNC     YES-RUB OUT OR LINE FEED
17476  002725 -        JMP     @XSTART     LINE FEED- START OVER
17477  010631 -        ISZ     AONU        RUB-IS IT 1ST CHAR.OF CONTS
17500  002722 -        JMP     @XREAD      YES-READ SAME CHAR.AGAIN
17501  014644 -  B     DSZ     AOW         NO-DECREMENT AOW
17502  000401 +        JMP     .+1
17503  126420          SUBZ    1,1         CREATE 0
17504  044626 -        STA     1,NUM       STORE 0 IN NUM
17505  030636 -        LDA     2,HOLD      RETRIEVE LAST WORD
```

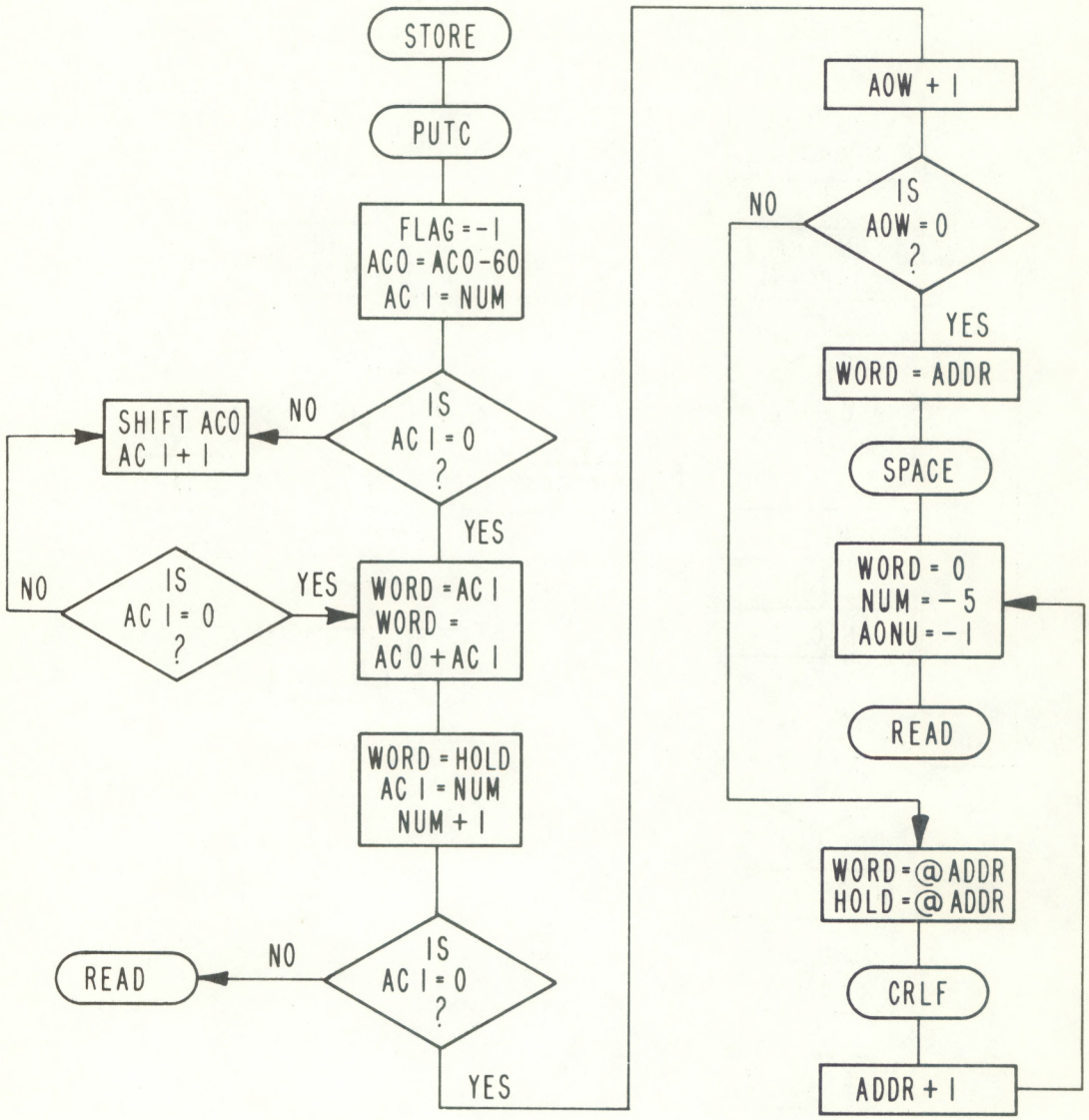
```
17506  000415 +          JMP     NOLTR       GO TO ERASE CHARACTER
17507  010621 -  C       ISZ     AONU        IS IT 1ST CHAR.OF CONTS.
17510  002712 -          JMP     @XREAD      YES-READ CHAR. AGAIN
17511  014623 -          DSZ     ADDR        NO-RESTORE LAST ADDRESS
17512  101213            MOVR#   0,0,SNC     IS IT RUB OUT OR LINE FD
17513  002475 +          JMP     @XCONS      LINE FD-START NEW WORD
17514  000765 -          JMP     B           RUB-PREPARE TO ERASE CHAR.
17515  101213     A      MOVR#   0,0,SNC     IS IT RUB OUT OR LINE FEED
17516  002472 +          JMP     @XCONS      LINE FD-START NEW WORD
17517  010611 -  AA      ISZ     AONU        RUB-INCREMENT AONU
17520  000401 +          JMP     .+1         DECREMENT NUM
17521  014611 -          DSZ     NUM
17522  030611 -          LDA     2,WORD
17523  020621 -  NOLTR   LDA     0,MSK
17524  024606 -          LDA     1,NUM       AC1 = NUM
17525  125005            MOV     1,1,SNR     IS AC1 = ZERO
17526  000406 +          JMP     .+6         YES-DO NOT SHIFT
17527  101120     SHIFT  MOVZL   0,0         NO-SHIFT MASK
17530  101120            MOVZL   0,0
17531  101120            MOVZL   0,0
17532  125404            INC     1,1,SZR     INC. AC1 . IS AC1 = ZERO
17533  000774 -          JMP     .-4         NO-SHIFT MASK AGAIN
17534  113400            AND     0,2         YES-ERASE CHARACTER
17535  052454 +          STA     2,@XWORD
17536  002664 -          JMP     @XREAD      READ NEXT CHARACTER
17537  152000     PRW    ADC     2,2         CREATE -1
17540  052452 +          STA     2,@XAOW     SET AOW = -1
17541  030433 +          LDA     2,ADIN      LOAD THE INSTRUCTION
17542  050402 +          STA     2,EXTR      PLACE INSTR.IN EXTR
17543  024646 -          LDA     1,CADR
17544  000000     EXTR                       TTH ADDRESS /  HTH CONTENTS
17545  020442 +          LDA     0,K60
17546  146443            SUBO    2,1,SNC     (SUBTRACT OUT DIGIT)
17547  101401            INC     0,0,SKP     INCR. 60 IF REMAINDER 0,+
17550  147001            ADD     2,1,SKP     RESTORE WORD
17551  000775 -          JMP     .-3
```

```
17552  004451 +          JSR     PUTC        PRINT DIGIT
17553  010771 -          ISZ     EXTR
17554  151203            MOVR    2,2,SNC     IS IT LAST DIGIT
17555  000767 -          JMP     EXTR        NO-NEXT DIGIT
17556  012434 +          ISZ     @XAOW       YES-INCR.AOW, IS AOW=ZERO
17557  000403 +          JMP     .+3         NO-   WORD IS FINISHED
17560  014631 -          DSZ     CADR        YES-   ADDRESS COMPLETE
17561  004432 +          JSR     SPP         PRINT SPACES
17562  030413 +    NWO   LDA     2,WOIN      LOAD THE INSTRUCTION
17563  050761 -          STA     2,EXTR      PLACE THE INSTR.IN EXTR
17564  004427 +          JSR     SPP         PRINT SPACES
17565  010624 -          ISZ     CADR        INCR. ADDRESS
17566  000401            JMP     .+1
17567  014623 -          DSZ     COUNT       DECR.COUNT, IS COUNT=ZERO
17570  000402 +          JMP     .+2         NO-OBTAIN CONTENTS
17571  002625 -          JMP     @JOUT       YES-END OF WORD
17572  026617 -          LDA     1,@CADR     LOAD CONTS. OF ADDRESS
17573  000751 -          JMP     EXTR
17574  030403       ADIN LDA     2,TTH
17575  030401       WOIN LDA     2,HTH
17576  100000       HTH                      HUNDRED THOUSAND
17577  010000       TTH                      TEN THOUSAND
17600  001000
17601  000100
17602  000010
17603  000001
17604  000012       C12                      ASCII(LINE FEED)
17605  000015       C15                      ASCII(CARR. RETURN)
17606  000040       K40                      ASCII(SPACE)
17607  000060       K60                      NEEDED TO BREAK DOWN ASCII
17610  017232 +     XCONS       A(CONS)      ADDRESS OF CONS
17611  017333 +     XWORD       A(WORD)      ADDRESS OF WORD
17612  017345 +     XAOW        A(AOW)       ADDRESS OF AOW
17613  030602 -     SPP   LDA   2,NSP        LOAD NUMBER OF SPACES
17614  020772 -     SPACE LDA   0,K40
-17615 063511            SKPBZ   TTO         WAIT UNTIL NOT BUSY
```

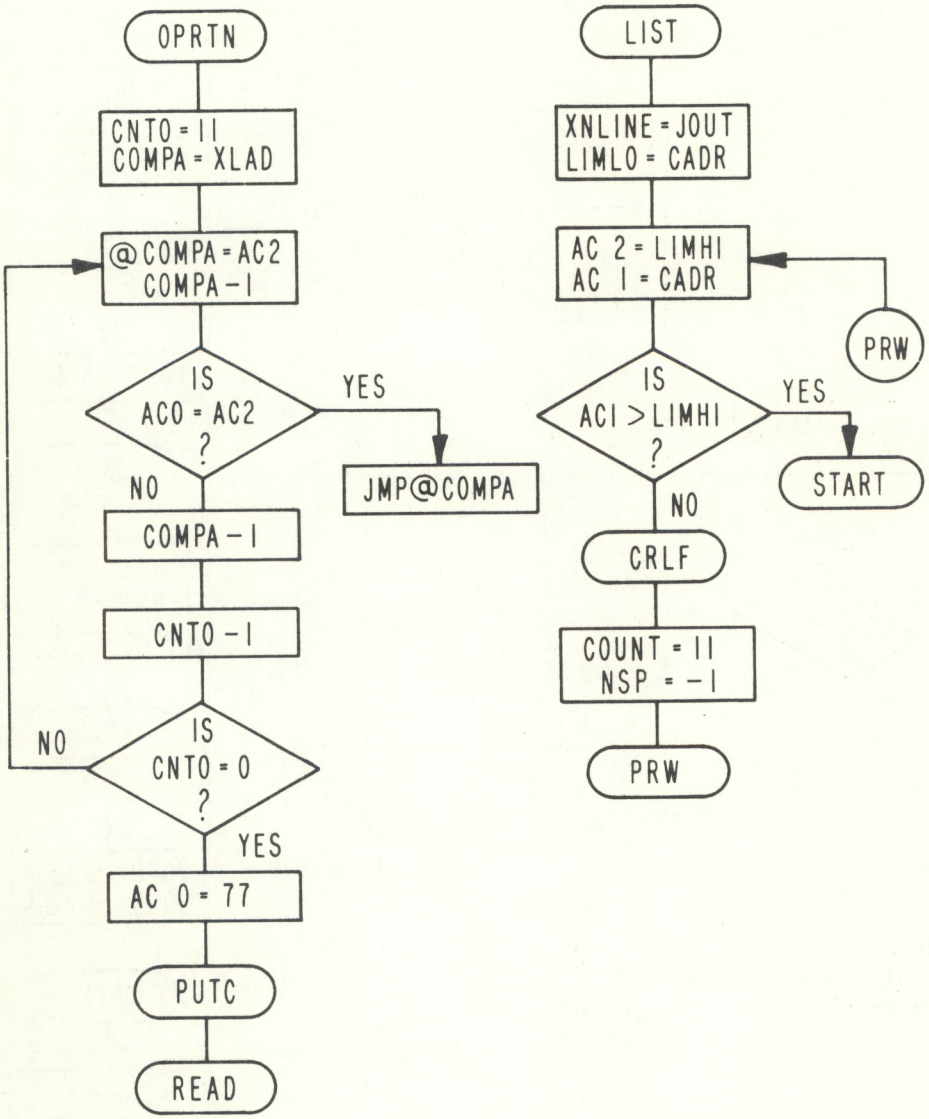
```
 17615  000777 -          JMP     .-1
-17617  061111            DOAS    0,TTO          OUTPUT CHARACTER(SPACE)
 17620  151404            INC     2,2,SZR        REDUCE SPACES REMAINING
 17621  000773 -          JMP     .-5            DO AGAIN
 17622  001400            JMP     0,3            GO BACK TO MAIN PROGRAM
-17623  063511    PUTC    SKPBZ   TTO            WAIT UNTIL NOT BUSY
 17624  000777 -          JMP     .-1
-17625  061111            DOAS    0,TTO          OUTPUT CHARACTER
 17626  101004            MOV     0,0,SZR        SKIP IF NULL CHAR.
 17627  001400            JMP     0,3            NULL-RETURN
 17630  171000    CRLF    MOV     3,2            SAVE RETURN ADDR.IN AC2
 17631  020754 -          LDA     0,C15
 17632  004771 -          JSR     PUTC           RETURN CARRIAGE
 17633  020751 -          LDA     0,C12
 17634  004767 -          JSR     PUTC           ADVANCE PAPER
 17635  001000            JMP     0,2            RETURN TO MAIN PROGRAM
```

APPENDIX D.

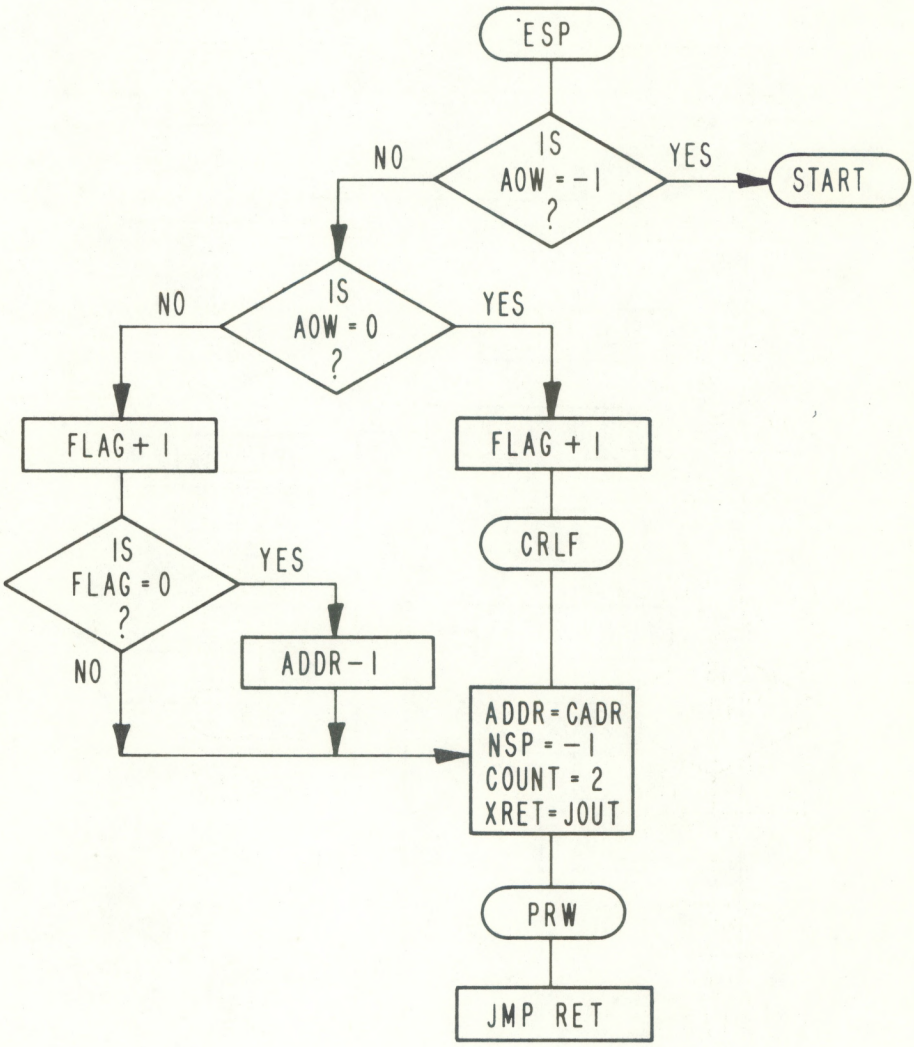
Flow Charts



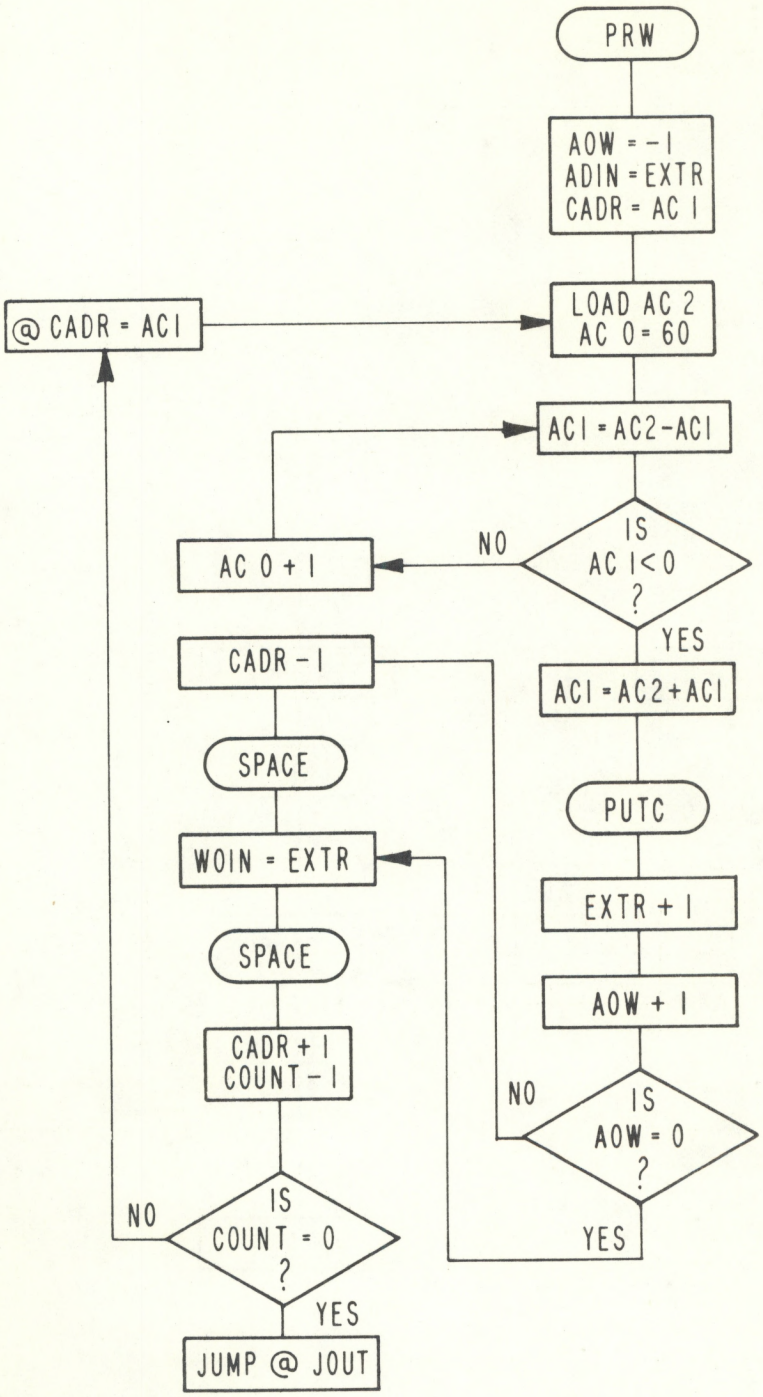
Storing an Address or Word



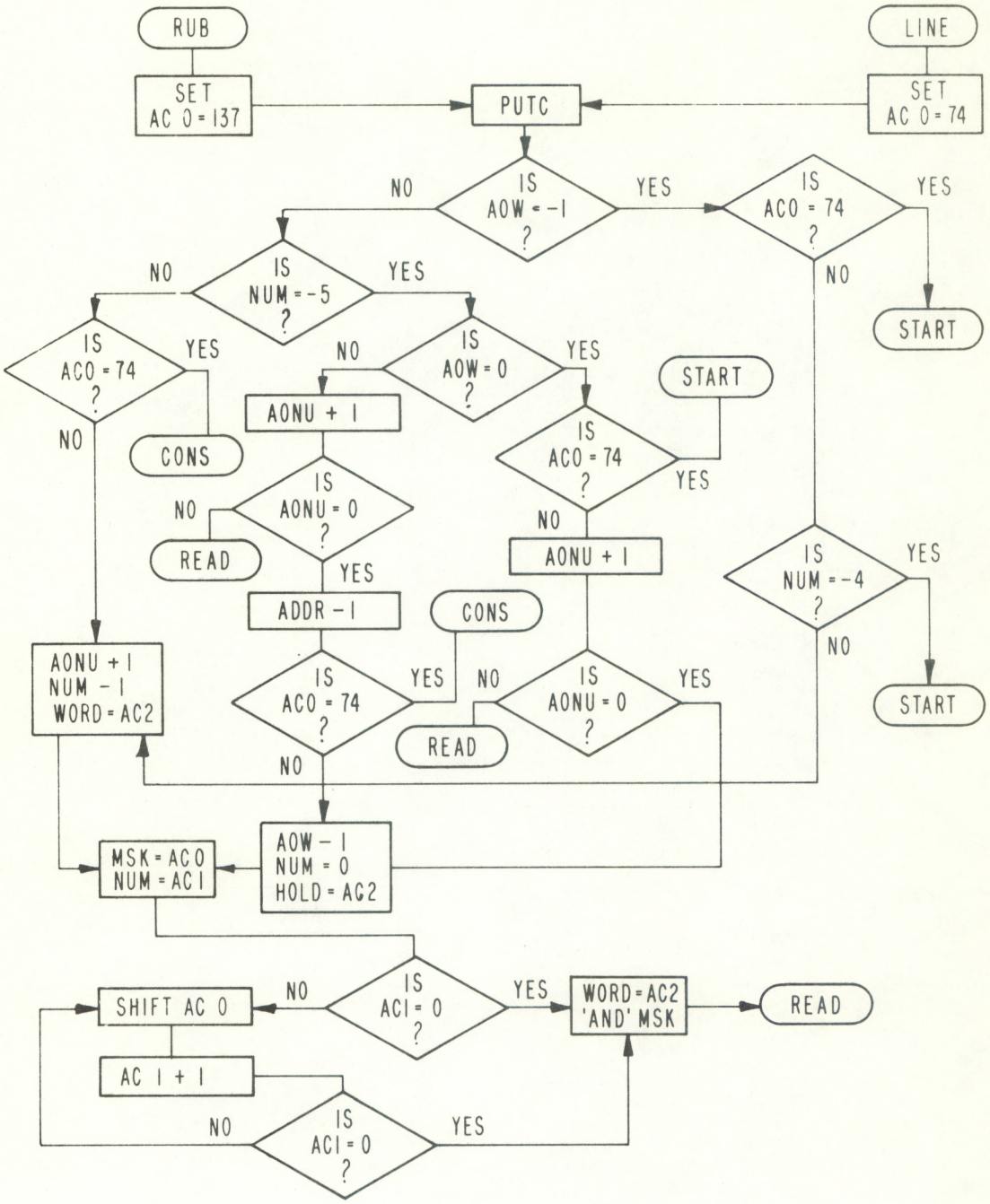
Selecting an Operation (Left); Initial Steps of a Dump (Right)



Obtaining Contents of a Location



Printing Contents of Location



Erasing a Word or Character

APPENDIX E

Sequence of Events

Event			Time→				
Key	ESC	1	6	2	7	3	
AOW	177777					000000	
NUM	177774	177775	17777.6	177777	000000	000001	177773
WORD	000000	10000	16000	16200	16270	16273	000000
ADDR						16273	
PRINT	CR/LF	1	6	7	7	3	SP
FLAG		000000	000000	000000	000000	000000	
AONU							177777
HOLD		10000	16000	16200	16270	16273	

KEY	1		4	1	
AOW				000001	
NUM	177774		000000	000001	177773
WORD	100000	(3,6,7)	136740	136741	000000
ADDR					16274
PRINT	1		4	1	CR/LF
FLAG	000000		000000	000000	
AONU					177777
HOLD	100000		136740	136741	136741