DATA GENERAL
CORPORATION
Southboro,
Massachusetts 01772
(617) 485-9100

PROGRAM

Exerciser

TAPES

Binary: 095-000012-02

ABSTRACT

Exerciser is a maintenance program designed to test the NOVA processor and paper tape equipment under conditions similar to actual customer usage.

097-000004-02

```
;                       EXERCISER

;1.     ABSTRACT
;              EXERCISER IS A MAINTENCE PROGRAM DESIGNED
;              TO TEST FOR RELIABLE OPERATION OF THE PROCESSOR
;              INSTRUCTIONS AND THE PAPER TAPE EQUIPMENT. THE
;              PROGRAM MAY EXERCISE THE TELETYPE READER/
;              PUNCH, HIGH SPEED READER/PUNCH, THE REAL TIME
;              CLOCK, AND THE NOVA INSTRUCTIONS. THE DEVICES
;              TO BE USED ARE SELECTED BY CONSOLE SWITCHS
;              AND ARE SERVICED VIA THE INTERRUPT SYSTEM.

;2.     MACHINE REQUIREMENTS
;2.1            STANDRED NOVA PROCESSOR
;2.2            2K READ/WRITE MEMORY, MINIMUM
;2.3            OPTIONAL EQUIPMENT
;2.3.1                  TELETYPE READER
;2.3.2                  TELETYPE PUNCH
;2.3.3                  HIGH SPEED READER
;2.3.4                  HIGH SPEED PUNCH
;2.3.5                  REAL TIME CLOCK

;3.     SWITCH SETTINGS
;3.1            STARTING ADDRESS =000002
;3.2            SWITCH 0(1)     =ACTIVATE HIGH SPEED PUNCH
;3.3            SWITCH 1(1)     ="         "    "      READER
;3.4            SWITCH 2(1)     ="         TELETYPE READER
;3.5            SWITCH 3(1)     ="         "         PUNCH
;3.6            SWITCH 4(1)     ="         REAL TIME CLOCK
;3.7            SWITCH 5(1)     =ACTIVATE SECOND PTP
;3.8            SWITCH 6(1)     =ACTIVATE SECOND PTR
;3.9            SWITCH 7(1)     =ACTIVATE SECOND TTI
;3.10           SWITCH 8(1)     =ACTIVATE SECOND TTO

;4.     OPERATING PROCEEDURE
;4.1            LOAD THE PROGRAM VIA THE BINARY LOADER
;4.2            SET SWITCHES TO 000002
;4.3            PRESS START
;4.4            THE PROGRAM WILL RUN UNTILL MANUALLY STOPPED
;              OR A ERROR IS DETECTED.
;4.5            IN/OUT EQUIPMENT TESTING
;4.5.1                  BY SETTING SWITCHES 0-4 THE CORRESPONDIN
;                       DEVICE WILL BE ACTIVATED. THE SWITCHS AR
;                       READ AT THE COMPLETION OF EACH PASS OF T
;                       INSTRUCTION TEST. IF A CONSOLE SWITCH HA
;                       BEEN SET/CLEARED DURING THE INSTRUCTION
;                       TEST THE DEVICE WILL BE ACTIVATED/DEACTI
;                       AT THE END OF A PASS.
;4.5.2                  TO TEST THE HIGH SPEED READER AND PUNCH
;4.5.2.1                        RAISE THE READER LEVER
;4.5.2.2                        SET SWITCHES 0 AND 1
;4.5.2.3                        WHEN 5 TO 6 FEET OF TAPE HAVE
;                               BEEN PUNCHED PRESS STOP.
;4.5.2.4                        INSERT THE TAPE INTO THE READER
;4.5.2.5                        CLOSE THE LEVER
;4.5.2.6                        PRESS CONTINUE
;4.5.2.7                        THE TAPE WILL BE CHECKED
```

```
;4.5.3                    TO TEST THE TELETYPE READER AND PUNCH
;4.5.3.1                        SET THE TELETYPE READER TO FREE
;4.5.3.2                        PRESS THE PUNCH ON BUTTON
;4.5.3.3                        SET SWITCHES 2 AND 3
;4.5.3.4                        WHEN SEVERAL INCHES OF TAPE
;                               HAVE BEEN PUNCHED INSERT THE
;                               LEADER... INTO THE READER.
;4.5.3.5                        SET THE READER SWITCH TO START
;4.5.3.6                        THE TAPE WILL BE CHECKED

;5.      PROGRAM OUTPUT/ERROR DISCRIPTION
;5.1             WHEN A MALFUNCTION IS DETECTED THE PROGRAM
;               WILL HALT. CONSULT THE LISTING FOR THE
;               CAUSE OF ERROR.
;5.2             UPON CONTINUING FROM A ERROR MOST ROUTINES
;               WILL ENTER A LOOP USING THE FAILING OPERANDS.
;5.3             MOST ROUTINES ISSUE A "P" PULSE(A74) FOR
;               SYNCING A SCOPE. IF POSSIBLE TRY TO OBSERVE
;               THE ERROR WITHOUT THE I/O EQUIPMENT RUNNING.

;6.      PROGRAM DISCRIPTION/THEORY OF OPERATION
;6.1             THE EXERCISER PROGRAM IS DESIGNED TO PRODUCE
;               A INTERNAL MACHINE ENVIRONMENT AS SIMULAR
;               TO NORMAL OPERATING CONDITIONS AS POSSIABLE.
;               THE PROGRAM ISSUES A I/O RESET PULSE AND EXECUTE
;               THE INSTRUCTION TEST. THE  INSTRUCTION TEST USES
;               A DATA BUFFER WHOS START AND STOP ADDRESS ARE
;               LOADED AS 3000 AND 3600. AT THE END OF EACH
;               PASS THE BUFFER START AND STOP PARAMETERS ARE
;               ADJUSTED TO THE MAXIMUM MEMORY SIZE. THE
;               INSTRUCTIONS WILL BE TESTED WITH THE SYSTEM
;               MEMORY SIZE ON SUCESSIVE PASSES. THE PROGRAM
;               WILL NOT USE THE LAST 200 LOCATIONS OF THE
;               HIGHEST MEMORY MODULE (LOCATION OF LOADERS).
;6.2             AT THE END OF EACH PASS THE CONSOLE SWITCHS
;               ARE READ. IF SWITCHS 0-8 ARE ALL ZERO THE
;               PROGRAM WILL RETURN TO CYCLE THE INSTRUCTION
;               TEST. IF A SWITCH HAS BEEN SET THE CORRESPOND-
;               ING DEVICE WILL BE ACTIVATED, AND THE INTERRUPT
;               SYSTEM TURNED ON. AFTER THE SWITCHS HAVE BEEN
;               INTERROGATED AND THE MEMORY SIZE DETERMINED THE
;               PROGRAM RETURNS TO THE INSTRUCTION TEST FOR
;               ANOTHER PASS.
;6.3             THE DEVICES ARE SERVICED VIA THE INTERRUPT
;               SYSTEM. THE "INTA" INSTRUCTION IS USED TO
;               DISPATCH TO THE ROUTINE REQUESTING SERVICE.
;               IF THE INTA READS BACK A IMPROPER DEVICE CODE
;                A HALT WILL OCCUR. WHEN DEVICES
;               ARE ACTIVE A HALT OCCURING WITH "ION" LIGHT OFF
;               INDICATES A ERROR AT THE INTERRUPT SERVICE LEVEL
```

^^^

```
;6.4            THE REAL TIME CLOCK IS ACTIVATED BY
;               SWITCH 4. THE CLOCK IS SET TO INTERRUPT
;               AT A 1000 HERTZ RATE. WHEN A CLOCK INTERRUPT
;               OCCURS THE BUSY AND DONE FLAGS ARE TESTED
;               BEFORE AND AFTER THE CLOCK IS RESTARTED.
;6.5            THE TELETYPE PUNCH IS ACTIVATED BY
;               SWITCH 3. THE PUNCH WILL PUNCH SEVERAL
;               INCHES OF LEADER FOLLOWED BY NON
;               ZERO,EVEN PARITY,RANDOM CHARACTORS.
;               THE PUNCHING WILL CONTINUE UNTILL
;               SWITCH 3 IS DEPRESSED, AT WHICH TIME
;               TRAILER WILL BE PUNCHED.
;6.6            THE TELETYPE READER IS ACTIVATED BY
;               SWITCH 2. THE READER WILL ACCEPT TAPE
;               PUNCHED EITHER ON THE TELETYPE OR HIGH
;               SPEED PUNCH. THE READER WILL INITIALIZE
;               ITS RANDOM NUMBER GENERATOR ON A ZERO CHAR-
;               ACTOR. IT IS NECESSARY TO LOAD THE READER IN
;               THE LEADER SECTION OF THE TAPE.
;6.7            THE HIGH SPEED READER IS ACTIVATED BY SWITCH
;               1. THE READER WILL READ AND CHECK TAPES
;               PRODUCED ON EITHER THE TELETYPE OR HIGH
;               SPEED PUNCH. IF THE PUNCH IS ACTIVE
;               WHEN TAPES ARE BEING READ THE READER WILL BE
;               SYNCRONIZED WITH THE PUNCH SUCH THAT THE TAPE
;               BEING PUNCHED MAY BE LOADED INTO THE READER.
;               IF THE HIGH SPEED PUNCH IS NOT ACTIVE A PRE-
;               PUNCHED TAPE WILL BE READ AT HIGH SPEED.
;6.8            THE HIGH SPEED PUNCH IS ACTIVATED BY SWITCH 0.
;               THE PUNCH WILL PUNCH 256 LINES OF LEADER
;               FOLLOWED BY NON ZERO,EVEN PARITY,RANDOM
;               CHARACTORS. PUNCHING WILL CONTINUE UNTILL
;               SWITCH 0 IS DEPRESSED, AT WHICH TIME TRAILER
;               WILL BE PUNCHED.

;7.      RESTRICTIONS/MISC
;7.1            BOTH THE TELETYPE AND HIGH SPEED READER MUST
;               BE IN THE LEADER SECTION OF THE TAPE WHEN
;               THAY ARE STARTED.
;7.2            SWITCHES ARE READ ONLY AT THE END OF A PASS,
;               THEREFOR, THERE IS A DELAY BETWEEN THE SWITCH
;               ACTION AND THE DEVICE RESPONSE.
;7.3            DEBUGING IS DIFFICULT IN THE ENVIORMENT
;               CREATED BY ACTIVE I/O EQUIPMENT. WHENEVER
;               POSSIABLE USE THE DIAGNOSTIC FOR THE
;               PERTICULAR I/O DEVICE FAILING.
;7.4            PRESSING A TELETYPE KEY MAY CAUSE A ERROR.
;7.5            READING A CHARACTOR OF ZEROS WILL
;               INITIALIZE THE RANDOM NUMBER TO 53 OCTAL.
;               THIS IS A IMPORTANT NUMBER IF PAPER TAPE
;               READERS ARE FAILING BY READING ALL ZEROS!
```

```
                    ;PAGE ZERO STUFF
       000001               .LOC 1
00001  000355               INTR
00002  002003               JMP @.+1
00003  000156               BEG
00004  063077               HALT                ;INTERRUPT FROM PLT,CDR,DIS
00005  000004               JMP .-1
       000040               .LOC 40

00040  000531  ITAB:        LSR                 ;KEYBOARD INTERRUPT
00041  000610               TYO                 ;TTO
00042  000510               HSR1                ;READER
00043  000410               PUN                 ;PUNCH
00044  000644               CLOCK
00045  000004               4
00046  000004               4
00047  000004               4
00050  000775               .LSR
00051  001034               .TYO
00052  000754               .HSR1
00053  000651               .PUN

00054  000000  LSRB:        0                   ;KEYBOARD CHAR BUFFER
00055  000000  LSRCH:       0                   ;"          "
00056  000000  KRAN:        0                   ;"          RANDOM
00057  000000  HSRB:        0                   ;READER CHAR BUFFER
00060  000000  HSRCH:       0                   ;"          "
00061  000000  READER:      0                   ;"          CHAR COUNTER
00062  000000  RRAN:        0                   ;"          RANDOM
00063  000000  PRAN:        0                   ;PUNCH RANDOM
00064  000000  OOT:         0                   ;OUT OF TAPE FLAG
00065  000377  LT:          377                 ;LEADER TRAILER
00066  000000  PUNCH:       0                   ;PUNCH CHAR COUNT
       000070               .LOC .+1
00070  000377  RPDIF:       377                 ;TAPE LENGTH BETWEEN
00071  000000  C170:        0
00072  000000  ACTION:      0                   ;ACTIVE DEVICES
00073  135525  C13552:      135525
00074  000000  RER:         0
00075  000177  C177:        177
00076  000010  C10:         10

00077  000565  CXRAND:      XRAND
00100  000000  ISAV0:       0
00101  000000  ISAV1:       0
00102  000000  ISAV2:       0
00103  000000  ISAV3:       0
00104  000000  ISAVC:       0
00105  000000  TRAN:        0
00106  000000  TYPE:        0
00107  000000  TEM:         0
00110  177773  CM5:         -5
00111  000004  C4:          4
```

^^^

```
00112 003777  MSIZE:  3777
00113 000040  C40:    40
00114 003140  BUFF:   3140
00115 003600  FIN:    3600
00116 003400  FIN200: 3400
00117 003340  BUF200: 3340
00120 000003  C3:     3
00121 000005  C5:     5
00122 000377  C377:   377
00123 000000  CFOO:   0
00124 177600  C174X:  177600
00125 001070  MAIN:   STA0
00126 003000  C3000:  3000
      000130          .LUC .+1
00130 001000  C1000:  1000
00131 000200  C200:   200
00132 000000  STP:    0
00133 000000  STT:    0
00134 000000  TIN:    0
00135 000000  TINEY:  0


00136 000037  C37:    37
00137 000000  .STP:   0
00140 000000  .RKAN:  0
00141 000000  .PRAN:  0
00142 000000  .PUNCH: 0
00143 000000  .READ:  0
00144 000000  .HSRB:  0
00145 000000  .HSRCH: 0
00146 000000  .TINEY: 0
00147 000000  .TIN:   0
00150 000000  .LSRB:  0
00151 000000  .LSRCH: 0
00152 000000  .KKAN:  0
00153 000000  .TYPE:  0
00154 000000  .TRAN:  0
00155 000000  .STT:   0

      000053  .PTP=53
      000052  .PTR=52
      000050  .TTI=50
      000051  .TTO=51
```

^^^

```
00156 102400  BEG:    SUB 0,0          ;STARTS HERE
00157 040072          STA 0,ACTION
00160 062677          IORST
00161 002125          JMP @MAIN        ;PROCESSOR TEST

00162 030072  BEG1:   LDA 2,ACTION
00163 064477          READS 1
00164 150000          COM 2,2
00165 133502          ANDL 1,2,SZC
00166 004276          JSR PTPSU        ;START PUNCH
00167 151102          MOVL 2,2,SZC
00170 004307          JSR PTRSU        ;START READER
00171 151102          MOVL 2,2,SZC
00172 004315          JSR TTISU        ;START TTI READER
00173 151102          MOVL 2,2,SZC
00174 004267          JSR TTOSU        ;START TTO OUTPUT
00175 151102          MOVL 2,2,SZC
00176 004264          JSR RTCSU        ;START REAL TIME CLOCK
00177 151102          MOVL 2,2,SZC     ;SECOND PUNCH
00200 004340          JSR .TPSU
00201 151102          MOVL 2,2,SZC     ;SECOND READER
00202 004351          JSR .TRSU
00203 151102          MOVL 2,2,SZC     ;SECOND TTI
00204 004324          JSR .TISU
00205 151102          MOVL 2,2,SZC     ;SECOND TTO
00206 004331          JSR .TOSU

00207 030072          LDA 2,ACTION
00210 044072          STA 1,ACTION
00211 124000          COM 1,1
00212 133502          ANDL 1,2,SZC
00213 004301          JSR PTPSD        ;STOP PUNCH
00214 151102          MOVL 2,2,SZC
00215 004305          JSR PTRSD
00216 151102          MOVL 2,2,SZC
00217 004313          JSR TTISD        ;STOP TTI READER
00220 151102          MOVL 2,2,SZC
00221 004272          JSR TTOSD        ;STOP TTO
00222 151102          MOVL 2,2,SZC
00223 060214          NIOC RTC         ;STOP CLOCK
00224 151102          MOVL 2,2,SZC
00225 004343          JSR .TPSD        ;STOP SECOND PTP
00226 101001          MOV 0,0,SKP
00227 000000          0
00230 151102          MOVL 2,2,SZC
00231 004347          JSR .TRSD        ;STOP SECOND PTR
00232 151102          MOVL 2,2,SZC
00233 004322          JSR .TISD        ;STOP SECOND TTI
00234 151102          MOVL 2,2,SZC
00235 004334          JSR .TOSD        ;STOP SECOND TTO
```

```
^^^
00236 024126  MEMSZ:  LDA 1,C3000      ;SIZE THE MEMORY
00237 020130          LDA 0,C1000
00240 107000          ADD 0,1
00241 044112          STA 1,MSIZE
00242 125112          MOVL# 1,1,SZC
00243 000250          JMP .+5
00244 046112          STA 1,@MSIZE
00245 032112          LDA 2,@MSIZE
00246 132415          SUB# 1,2,SNR
00247 000237          JMP MEMSZ+1
00250 014112          DSZ MSIZE         ;MEMORY MASK
00251 020131          LDA 0,C200
00252 024112          LDA 1,MSIZE
00253 106400          SUB 0,1
00254 044115          STA 1,FIN
00255 106400          SUB 0,1
00256 044116          STA 1,FIN200
00257 020124          LDA 0,C174X
00260 024072          LDA 1,ACTION
00261 107404          AND 0,1,SZR       ;IF SWITCHS SET TURN
00262 060177          NIOS CPU          ;ON INTERRUPT ENABLE.
00263 002125          JMP @MAIN


00264 020120  RTCSU:  LDA 0,C3          ;START REAL TIME CLOCK
00265 061114          DOAS 0,RTC
00266 001400          JMP 0,3

00267 102400  TTOSU:  SUB 0,0
00270 061111          DOAS 0,TTO
00271 102401          SUB 0,0,SKP
00272 102000  TTOSD:  ADC 0,0           ;TELETYPE PUNCH
00273 040106          STA 0,TYPE
00274 040133          STA 0,STT
00275 001400          JMP 0,3
00276 102400  PTPSU:  SUB 0,0           ;HIGH SPEED PUNCH
00277 061113          DOAS 0,PTP
00300 102401          SUB 0,0,SKP
00301 102000  PTPSD:  ADC 0,0
00302 040066          STA 0,PUNCH
00303 040132          STA 0,STP
00304 001400          JMP 0,3

00305 060212  PTRSD:  NIOC PTR         ;READER
00306 101011          MOV# 0,0,SKP
00307 060112  PTRSU:  NIOS PTR
00310 102400          SUB 0,0
00311 040061          STA 0,READER
00312 001400          JMP 0,3

00313 010135  TTISD:  ISZ TINEY
00314 001400          JMP 0,3
00315 060110  TTISU:  NIOS TTI
00316 102400          SUB 0,0
00317 040134          STA 0,TIN
00320 040135          STA 0,TINEY
00321 001400          JMP 0,3
```

^^^

```
00322 010146  .TISD:  ISZ .TINEY
00323 001400          JMP 0,3          ;SECOND TTI
00324 060150  .TISU:  NIOS .TTI
00325 102400          SUB 0,0
00326 040147          STA 0,.TIN
00327 040146          STA 0,.TINEY
00330 001400          JMP 0,3

00331 102400  .TOSU:  SUB 0,0          ;SECOND TTO
00332 061151          DOAS 0,.TTO
00333 102401          SUB 0,0,SKP
00334 102000  .TOSD:  ADC 0,0
00335 040153          STA 0,.TYPE
00336 040155          STA 0,.STT
00337 001400          JMP 0,3

00340 102400  .TPSU:  SUB 0,0          ;SECOND PTP
00341 061153          DOAS 0,.PTP
00342 102401          SUB 0,0,SKP
00343 102000  .TPSD:  ADC 0,0
00344 040142          STA 0,.PUNCH
00345 040137          STA 0,.STP
00346 001400          JMP 0,3

00347 060252  .TRSD:  NIOC .PTR        ;SECOND PTR
00350 101011          MOV# 0,0,SKP
00351 060152  .TRSU:  NIOS .PTR
00352 102400          SUB 0,0
00353 040143          STA 0,.READER
00354 001400          JMP 0,3
```

```
^^^
00355 040100  INTR:   STA 0,ISAV0      ;INTERRUPT DISPATCH
00356 044101          STA 1,ISAV1      ;SAVE INTERRUPTS
00357 050102          STA 2,ISAV2
00360 054103          STA 3,ISAV3
00361 175200          MOVR 3,3
00362 054104          STA 3,ISAVC
00363 061477          INTA 0
00364 030136          LDA 2,C37
00365 113520          ANDZL 0,2,
00366 024113          LDA 1,C40
00367 107620          ANDZR 0,1
00370 133220          ADDZR 1,2
00371 024076          LDA 1,C10
00372 034415          LDA 3,C24
00373 172433          SUBZ# 3,2,SNC
00374 132423          SUBZ 1,2,SNC
00375 063077          HALT             ;UNKOWN INTERRUPT?
00376 003040          JMP @ITAB,2      ;SERVICE THE DEVICE

00377 020104  DISMIS: LDA 0,ISAVC      ;RESTORE,EXIT INTERRUPT
00400 101100          MOVL 0,0
00401 020100          LDA 0,ISAV0
00402 024101          LDA 1,ISAV1
00403 030102          LDA 2,ISAV2
00404 034103          LDA 3,ISAV3
00405 060177          INTEN
00406 002000          JMP @0

00407 000024  C24:    24
00410 006077  PUN:    JSR @CXRAND      ;PUNCH INTERRUPT
00411 000063          PRAN
00412 024065          LDA 1,LT         ;LEADER/TRAILER
00413 030066          LDA 2,PUNCH      ;CHECK
00414 132032          ADCZ# 1,2,SZC
00415 000425          JMP PUN1
00416 020073          LDA 0,C135525    ;INIT RANDOM ON L/T
00417 040063          STA 0,PRAN
00420 024075          LDA 1,C177
00421 034132          LDA 3,STP
00422 133405          AND 1,2,SNR
00423 175005          MOV 3,3,SNR
00424 000415          JMP PUNX
00425 000403          JMP .+3
00426 000000          0
00427 000000          0

00430 060213          NIOC PTP         ;STOP PUNCHING
00431 063412          SKPBN PTR
00432 063712          SKPDZ PTR
00433 000377          JMP DISMIS
00434 020072          LDA 0,ACTION
00435 101100          MOVL 0,0
00436 101102          MOVL 0,0,SZC
00437 060112          NIOS PTR         ;RESTART READER
00440 000377          JMP DISMIS
00441 102400  PUNX:   SUB 0,0          ;PUNCH LEADER/TRAILER
```

^^^

```
00442 010066  PUN1:   ISZ PUNCH        ;CHAR COUNTER
00443 101001          MOV 0,0,SKP
00444 010066          ISZ PUNCH
00445 063413          SKPBN PTP
00446 063613          SKPDN PTP
00447 063077          HALT             ;PUNCH FLAG CHECK.
00450 061113          DOAS 0,PTP
00451 063513          SKPBZ PTP
00452 063713          SKPDZ PTP
00453 063077          HALT

00454 020066          LDA 0,PUNCH      ;WHEN PUNCHING AND
00455 024061          LDA 1,READER     ;READING IT MAY
00456 030070          LDA 2,RPDIF      ;BE NECESSARY TO
00457 106400          SUB 0,1          ;RESTART THE READER
00460 133404          AND 1,2,SZR
00461 000377          JMP DISMISS      ;EXIT INTERRUPT



00462 063412          SKPBN PTR        ;PUNCH ROUTINE CONT.
00463 063712          SKPDZ PTR
00464 000377          JMP DISMISS
00465 020072          LDA 0,ACTION
00466 101100          MOVL 0,0
00467 101112          MOVL# 0,0,SZC    ;OPERATOR REQUEST
00470 060112          NIOS PTR         ;READER,START IT.
00471 000377          JMP DISMISS

00472 064512  HSR:    DIAS 1,PTR       ;READER INTERRUPT
00473 020057          LDA 0,HSRB
00474 044057          STA 1,HSRB       ;CHAR BUFFER
00475 040060          STA 0,HSRCH      ;CHAR TO BE USED.
00476 030061          LDA 2,READER
00477 101004          MOV 0,0,SZR
00500 151235          MOVZR# 2,2,SNR
00501 000425          JMP HSR3         ;ZERO DATA OR FIRST CHAR.
00502 006077          JSR @CXRAND      ;GEN RANDOM CHAR
00503 000062          RRAN
00504 024060          LDA 1,HSRCH      ;CHECK DATA
00505 106414          SUB# 0,1,SZR
00506 063077          HALT             ;C(0)=GOOD CHARACTOR
00507 000377          JMP DISMIS       ;C(1)=BAD CHARACTOR.
```

^^^

```
00510 020072  HSR1:   LDA 0,ACTION
00511 010061          ISZ READER
00512 101001          MOV 0,0,SKP
00513 010061          ISZ READER
00514 101103          MOVL 0,0,SNC
00515 000755          JMP HSR
00516 020066          LDA 0,PUNCH
00517 024061          LDA 1,READER
00520 030070          LDA 2,RPDIF       ;TAPE LENGTH
00521 106400          SUB 0,1
00522 133404          AND 1,2,SZR       ;DON'T LET READER READ
00523 000747          JMP HSR           ;TO MUCH TAPE
00524 064612          DIAC 1,PTR
00525 000746          JMP HSR+1         ;READ AND STOP
00526 020073  HSR3:   LDA 0,C135525
00527 040062          STA 0,RRAN
00530 000377          JMP DISMIS        ;RANDOM SETUP



00531 020135  LSR:    LDA 0,TINEY       ;KEYBOARD
00532 101005          MOV 0,0,SNR
00533 000403          JMP .+3
00534 060210          NIOC TTI
00535 000377          JMP DISMIS        ;STOP READER
00536 064510          DIAS 1,TTI        ;KEYBOARD INTERRUPT
00537 063610          SKPDN TTI
00540 063410          SKPBN TTI
00541 063077          HALT              ;FLAG CHECK
00542 030134          LDA 2,TIN
00543 010134          ISZ TIN
00544 101001          MOV 0,0,SKP
00545 010134          ISZ TIN
00546 020054          LDA 0,LSRB
00547 044054          STA 1,LSRB        ;CHAR BUFFER
00550 040055          STA 0,LSRCH       ;THE CHARACTOR TO USE
00551 151004          MOV 2,2,SZR
00552 101005          MOV 0,0,SNR
00553 000407          JMP LSR1          ;ZERO CHARACTOR
00554 006077          JSR @CXRAND
00555 000056          KRAN
00556 024055          LDA 1,LSRCH       ;CHECK DATA
00557 106414          SUB# 0,1,SZR
00560 063077          HALT              ;C(1)=BAD,C(0)=GOOD
00561 000377          JMP DISMISS
00562 020073  LSR1:   LDA 0,C135525
00563 040056          STA 0,KRAN
00564 000377          JMP DISMISS
```

```
^^^
00565 027400  XRAND:  LDA 1,@0,3       ;GENERATE A 8 BIT
00566 020075          LDA 0,C177       ;EVEN PARITY RANDOM
00567 131120          MOVZL 1,2        ;NUMBER.
00570 151120          MOVZL 2,2
00571 133000          ADD 1,2
00572 024402          LDA 1,.+2
00573 147000          ADD 2,1
00574 101300          MOVS 0,0
00575 123725          ANDZS 1,0,SNR
00576 000770          JMP XRAND+1
00577 047400          STA 1,@0,3
00600 111300          MOVS 0,2
00601 126000          ADC 1,1
00602 107000          ADD 0,1
00603 123404          AND 1,0,SZR
00604 000775          JMP .-3
00605 101200          MOVR 0,0
00606 143300          ADDS 2,0         ;C(0)R=EVEN PARITY NUMBER.
00607 001401          JMP 1,3

00610 006077  TYO:    JSR @CXRAND      ;TELETYPE OUTPUT
00611 000105          TRAN
00612 024065          LDA 1,LT
00613 125220          MOVZR 1,1        ;LEADER/2
00614 030106          LDA 2,TYPE
00615 132032          ADCZ# 1,2,SZC
00616 000413          JMP TYO1
00617 020073          LDA 0,C135525
00620 040105          STA 0,TRAN       ;INIT RANDOM
00621 125220          MOVZR 1,1
00622 034133          LDA 3,STT
00623 133405          AND 1,2,SNR
00624 175005          MOV 3,3,SNR
00625 000403          JMP .+3
00626 060211          NIOC TTO         ;SHUT DOWN
00627 000377          JMP DISMIS
00630 102400          SUB 0,0

00631 010106  TYO1:   ISZ TYPE
00632 101001          MOV 0,0,SKP
00633 010106          ISZ TYPE
00634 063411          SKPBN TTO
00635 063611          SKPDN TTO
00636 063077          HALT             ;FLAG CHECK
00637 061111          DOAS 0,TTO
00640 063511          SKPBZ TTO
00641 063711          SKPDZ TTO
00642 063077          HALT             ;FLAG CHECK
00643 000377          JMP DISMIS

00644 063414  CLOCK:  SKPBN RTC        ;CLOCK INTERRUPT
00645 063614          SKPDN RTC
00646 063077          HALT
00647 060114          NIOS RTC
00650 000377          JMP DISMIS
```

^^^

```
00651 006077   .PUN:   JSR @CXRAND      ;PUNCH INTERRUPT
00652 000141           .PRAN
00653 024065           LDA 1,LT          ;LEADER/TRAILER
00654 030142           LDA 2,.PUNCH
00655 132032           ADCZ# 1,2,SZC
00656 000423           JMP .PUN1
00657 020073           LDA 0,C135525
00660 040141           STA 0,.PRAN
00661 024075           LDA 1,C177
00662 034137           LDA 3,.STP
00663 133405           AND 1,2,SNR
00664 175005           MOV 3,3,SNR
00665 000413           JMP .PUNX

00666 060253           NIOC .PTP         ;STOP PUNCHING
00667 063452           SKPBN .PTR
00670 063752           SKPDZ .PTR
00671 000377           JMP DISMIS
00672 020072           LDA 0,ACTIUN
00673 101300           MOVS 0,0
00674 101200           MOVR 0,0
00675 101202           MOVR 0,0,SZC
00676 060152           NIOS .PTR         ;RESTART READER
00677 000377           JMP DISMIS
00700 102400   .PUNX:   SUB 0,0          ;PUNCH LEADER/TRAILER

00701 010142   .PUN1:   ISZ .PUNCH       ;CHARACTOR COUNTER
00702 101001           MOV 0,0,SKP
00703 010142           ISZ .PUNCH
00704 063453           SKPBN .PTP
00705 063653           SKPDN .PTP
00706 063077           HALT              ;PUNCH FLAG CHECK
00707 061153           DOAS 0,.PTP
00710 063553           SKPBZ .PTP
00711 063753           SKPDZ .PTP
00712 063077           HALT

00713 020142           LDA 0,.PUNCH      ;WHEN PUNCHING AND
00714 024143           LDA 1,.READER     ;READING IT MAY
00715 030070           LDA 2,RPDIF       ;BE NECESSARY TO
00716 106400           SUB 0,1           ;RESTART THE READER
00717 133404           AND 1,2,SZR
00720 000377           JMP DISMISS       ;EXIT INTERRUPT

00721 063452           SKPBN .PTR
00722 063752           SKPDZ .PTR
00723 000377           JMP DISMIS
00724 020072           LDA 0,ACTION
00725 101100           MOVL 0,0
00726 101100           MOVL 0,0
00727 101100           MOVL 0,0
00730 101100           MOVL 0,0
00731 101100           MOVL 0,0
00732 101100           MOVL 0,0
00733 101112           MOVL# 0,0,SZC
00734 060152           NIOS .PTR         ;RESTART READER
00735 000377           JMP DISMISS
```

^^^

```
00736 064552  .HSR:   DIAS 1,.PTR      ;SECOND READER INTERRUPT
00737 020144          LDA 0,.HSRB
00740 044144          STA 1,.HSRB
00741 040145          STA 0,.HSRCH
00742 030143          LDA 2,.READER
00743 101004          MOV 0,0,SZR
00744 151235          MOVZR# 2,2,SNR
00745 000425          JMP .HSR3         ;FIRST CHAR OR ZERO DATA
00746 006077          JSR @CXRAND       ;GEN RANDOM CHAR
00747 000140          .RRAN
00750 024145          LDA 1,.HSRCH      ;CHECK DATA
00751 106414          SUB# 0,1,SZR      ;READER/PUNCH
00752 063077          HALT              ;SECOND DEVICES. C(0)=GOOD
00753 000377          JMP DISMIS        ;C(1)=BAD

00754 020072  .HSR1:  LDA 0,ACTION
00755 010143          ISZ .READER
00756 101001          MOV 0,0,SKP
00757 010143          ISZ .READER
00760 101103          MOVL 0,0,SNC
00761 000755          JMP .HSR
00762 020142          LDA 0,.PUNCH
00763 024143          LDA 1,.READER
00764 030070          LDA 2,RPDIF
00765 106400          SUB 0,1           ;DONT LET READER READ TO MUCH
00766 133404          AND 1,2,SZR
00767 000747          JMP .HSR
00770 064652          DIAC 1,.PTR
00771 000746          JMP .HSR+1
00772 020073  .HSR3:  LDA 0,C135525
00773 040140          STA 0,.RRAN
00774 000377          JMP DISMIS

00775 020146  .LSR:   LDA 0,.TINEY      ;SECOND KEYBOARD
00776 101005          MOV 0,0,SNR
00777 000403          JMP .+3
01000 060250          NIOC .TTI
01001 000377          JMP DISMIS        ;STOP SECOND READER
01002 064550          DIAS 1,.TTI
01003 063650          SKPDN .TTI
01004 063450          SKPBN .TTI
01005 063077          HALT
01006 030147          LDA 2,.TIN
01007 010147          ISZ .TIN
01010 101001          MOV 0,0,SKP
01011 010147          ISZ .TIN
01012 020150          LDA 0,.LSRB
01013 044150          STA 1,.LSRB       ;CHAR BUFFER
01014 040151          STA 0,.LSRCH      ;THE CHARACTOR TO USE
01015 151004          MOV 2,2,SZR
01016 101005          MOV 0,0,SNR
01017 000412          JMP .LSR1
01020 006077          JSR @CXRAND
01021 000152          .KRAN
```

```
AAA

01022 024151            LDA 1,.LSRCH
01023 106414            SUB# 0,1,SZR        ;SECOND TELETYPE
01024 063077            HALT                ;C(0)=GOOD,C(1)=BAD
01025 000377            JMP DISMIS

01026 000000            0
01027 000000            0
01030 000000            0
01031 020073  .LSR1:    LDA 0,C135525
01032 040152            STA 0,.KRAN
01033 000377            JMP DISMIS

01034 006077  .TYO:     JSR @CXRAND         ;SECOND TTY (OUTPUT)
01035 000154            .TRAN
01036 024065            LDA 1,LT
01037 125220            MOVZR 1,1
01040 030153            LDA 2,.TYPE
01041 132032            ADCZ# 1,2,SZC
01042 000413            JMP .TYO1
01043 020073            LDA 0,C135525
01044 040154            STA 0,.TRAN         ;INIT RANDOM
01045 125220            MOVZR 1,1
01046 034155            LDA 3,.STT
01047 133405            AND 1,2,SNR
01050 175005            MOV 3,3,SNR
01051 000403            JMP .+3
01052 060251            NIOC .TTO           ;SHUT DOWN
01053 000377            JMP DISMIS
01054 102400            SUB 0,0             ;LEADER/TRAILER

01055 010153  .TYO1:    ISZ .TYPE
01056 101001            MOV 0,0,SKP
01057 010153            ISZ .TYPE
01060 063451            SKPBN .TTO
01061 063651            SKPDN .TTO
01062 063077            HALT                ;FLAG CHECK
01063 061151            DOAS 0,.TTO
01064 063551            SKPBZ .TTO
01065 063751            SKPDZ .TTO
01066 063077            HALT                ;FLAG CHECK
01067 000377            JMP DISMIS
                .EOT
```

```
                ;A TEST OF STA,LDA
                ;LOAD C(1) FROM BUFFER, STORE C(1) IN BUFFER+1. THIS
                ;PROCEEDURE
                ;WILL FILL THE BUFFER WITH THE VALUE OF THE FIRST WORD
                ;IN THE
                ;BUFFER.  THE C(1) IS TESTED WHEN THE BUFFER HAS BEEN
                ;FILLED.
                ;BECAUSE THE LAST WORD IN THE BUFFER IS BASED ON THE
                ;PREVIOUS
                ;WORD ITS CORRECTNESS INSURES THAT THE ENTIRE BUFFER IS
                ;CORRECT.
                ;IF THE C(1) IS IN ERROR (NOT ZERO) EXAMINE THE BUFFER

01070 030114    STA0:   LDA 2,BUFF      ;FIRST LOCATION IN BUFFER
01071 034115            LDA 3,FIN       ;LAST LOCATION IN BUFFER
01072 102400            SUB 0,0         ;C(0)=0
01073 041000            STA 0,0,2       ;VALUE OF FIRST BUFFER WORD

01074 025000    STA01:  LDA 1,0,2       ;WORD FROM BUFFER
01075 045001            STA 1,1,2       ;PUT INTO BUFFER+1
01076 151400            INC 2,2         ;MOVE TO NEXT BUFFER LOC
01077 156014            ADC# 2,3,SZR    ;TEST FOR END OF BUFFER
01100 000774            JMP .-4         ;MORE WORDS TO OUT PUT
01101 125014            MOV# 1,1,SZR    ;FULL BUFF! CHECK C(1)=0
01102 063077            HALT            ;EXAMINE BUFFER FOR ERRORS
01103 101014            MOV# 0,0,SZR    ;C(0) WAS CLEARED AT STA0+2
01104 063077            HALT            ;NOT OTHERWISE USED!

                ;A TEST OF STA,LDA
                ;THIS TEST IS IDENTICAL TO THE PREVIOUS TEST EXCEPT THE
                ;FIRST
                ;WORD OF THE BUFFER IS SET TO ALL ONES (177777).

01105 030114    STA02:  LDA 2,BUFF      ;BUFF CONTAINS FIRST BUFFER LOC
01106 034115            LDA 3,FIN       ;THE LAST LOCATION IN BUFFER
01107 102000            ADC 0,0         ;C0)=177777
01110 041000            STA 0,0,2       ;VALUE OF FIRST BUFFER WORD

01111 025000    STA03:  LDA 1,0,2       ;WORD FROM BUFFER
01112 045001            STA 1,1,2       ;PUT INTO BUFFER+1
01113 151400            INC 2,2         ;ADVANCE TO NEXT LOCATION
01114 156014            ADC# 2,3,SZR    ;TEST FOR END OF BUFFER
01115 000774            JMP .-4         ;MORE TO FILL

01116 124014            COM# 1,1,SZR    ;CHECK C(1) FOR 177777.
01117 063077            HALT            ;EXAMINE C(BUFFER) FOR ERROR
01120 100014            COM# 0,0,SZR    ;C(0) WAS MODIFIED DURING
01121 063077            HALT            ;THIS TEST! ?
```

^^^

```
                        ;FILL AND CHECK BUFFER. A BUFFER OF ALTERNATING BITS

01122 030114  STA04:  LDA 2,BUFF        ;FIRST LOCATION OF THE BUFFER
01123 034115          LDA 3,FIN         ;LAST  LOCATION OF THE BUFFER
01124 020420          LDA 0,C52525
01125 104000          COM 0,1
01126 045000          STA 1,0,2         ;BUFFER = 125252
01127 041001          STA 0,1,2         ;BUFFER = 052525
01130 021000  STA05:  LDA 0,0,2         ;THIS PAIR OF WORDS
01131 151400          INC 2,2           ;IS MOVED THROUGH THE
01132 156015          ADC# 2,3,SNR      ;BUFFER
01133 000774          JMP .=4

01134 021000  STA06:  LDA 0,0,2         ;GET BUFFER'S LAST AND
01135 025377          LDA 1,-1,2        ;LAST-1 WORSDS
01136 123020          ADDZ 1,0          ;C(0) SHOULD=177777
01137 100014          COM# 0,0,SZR      ;BUFFER SHOULD CONTAIN
01140 063077          HALT              ;ALTERNATING WORDS
01141 101012          MOV# 0,0,SZC      ;EXAMINE BUFFER FOR ERRORS
01142 063077          HALT              ;HOW DID CARRY CHANGE?
01143 101011          MOV# 0,0,SKP      ;SKIP OVER CONSTANT
01144 052525  C52525: 052525

                        ;MAKE SURE THAT ALL LOCATIONS IN THE BUFFER EXIST.
                        ;STORE A ADDRESS PATTERN (EACH LOCATION CONTAINS ITS
                        ;ADDRESS).
                        ;IF STA WERE ATTEMPTING TO STORE A 2400 IN LOCATION
                        ;2400
                        ;BUT INSTEAD STORED IN 2000, THE CONTENTS OF 2000 WOULD
                        ;CONTAIN 2400.ONE SHOULD SUSPECT MA BIT 7 OR OTHER
                        ;MEMORY ADDRESS DECODING.

01145 102400  STA10:  SUB 0,0           ;C(0) NOT USED THIS TEST
01146 034114          LDA 3,BUFF        ;FIRST LOCATION OF BUFFER
01147 024115          LDA 1,FIN         ;LAST  LOCATION OF BUFFER

01150 055400  STA11:  STA 3,0,3         ;STORE THE ADDRESS AT ADDRESS
01151 031400          LDA 2,0,3         ;LOAD C(2) WITH DATA JUST
01152 156414          SUB# 2,3,SZR      ;STORED. C(3)=DATA SENT,
01153 063077          HALT              ;C(3)=ADDRESS,C(2)=RECEIVED
01154 175400          INC 3,3           ;INC TO NEXT ADDR
01155 136414          SUB# 1,3,SZR      ;TEST FOR END OF BUFF
01156 000772          JMP STA11

01157 034114  STA12:  LDA 3,BUFF        ;CORE SHOULD BE FILLED WITH
01160 031400          LDA 2,0,3         ;ADDRESS PATTERN. GET WORD
01161 156414          SUB# 2,3,SZR      ;C(2) SHOULD EQUAL C(3)
01162 063077          HALT              ;LOOK FOR PATTERN OF ERRORS
01163 175400          INC 3,3           ;GO TO NEXT ADDRESS
01164 136414          SUB# 1,3,SZR      ;TEST FOR END OF BUFFER
01165 000773          JMP STA12+1       ;MORE TO GO

01166 101014  STA13:  MOV# 0,0,SZR      ;CHANGE TO (JMP STA10) FOR LOOP
01167 063077          HALT              ;C(0) NOT USED THIS TEST!
```

```
AAA

                ;THIS IS A ADDRESS TEST IN WHICH EACH LOCATION IN THE
                ;BURFER
                ;CONTAINS THE COMPLEMENT OF ITS ADDRESS. IF A FAILURE
                ;OCCURES
                ;EXAMINE THE BUFFER AND TRY TO FIND A PATTERN OF ERRORS
                ;LDA AND STA HAVE PREVIOUSLY WORKED WITH A SIMULAR
                ;PATTERN.

01170 034114  STA20: LDA 3,BUFF      ;C(3)=FIRST BUFFER LOCATION
01171 024115         LDA 1,FIN       ;C(1)=LAST  BUFFER LOCATION

01172 160005  STA21: COM 3,0,SNR     ;THE CONTENTS OF 3 SHOULD
01173 063077         HALT            ;ALLWAYS PRODUCE NOT ZERO
01174 041400         STA 0,0,3       ;STORE LOGICAL COMPLEMENT
01175 031400         LDA 2,0,3       ;OF ADDRESS IN ADDRESS
01176 142414         SUB# 2,0,SZR    ;C(2) SHOULD BE SAME AS
01177 063077         HALT            ;C(0) WHICH WAS STORED.
01200 175400         INC 3,3         ;INC TO NEST ADDRESS
01201 136414         SUB# 1,3,SZR    ;TEST FOR END OF BUFFER
01202 000770         JMP STA21       ;NOT YET END OF BUFFER


01203 034114  STA22: LDA 3,BUFF      ;RESET C(3) TO FIRST LOCATION
01204 031400         LDA 2,0,3       ;IN THE BUFFER. WORD FROM
01205 140000         COM 2,0         ;BUFFER SHOULD BE COMP OF
01206 116414         SUB# 0,3,SZR    ;ADDRESS IN 3. C(2)=BUFFER WD
01207 063077         HALT            ;C(0)=ITS COMP. C(3)=ADDRESS
01210 175400         INC 3,3         ;INCREMENT TO NEST ADDRESS
01211 136414         SUB# 1,3,SZR    ;TEST FOR END OF BUFFER
01212 000772         JMP STA22+1     ;MORE WORDS TO CHECK

01213 101010         MOV# 0,0        ;REPLACE WITH (JMP STA20) TO
                                     ;LOOP
```

^^^

```
                ;A TEST OF JSR
                ;THIS PROGRAM WILL EXECUTE A JSR INSTRUCTION TO EVERY
                ;LOCATION
                ;IN THE BUFFER. EACH LOCATION IN THE BUFFER CONTAINS
                ;A(JSR 1,3)
                ;INSTRUCTION. THE JSR RETURN FROM THE BUFFER WILL BE TO
                ;THE
                ;CALLING (JSR TO BUFFER) +2. THE RETURN FROM THE BUFFER
                ;WILL
                ;ALSO STORE THE PC (PROGRAM COUNTER) OF THE BUFFER
                ;INSTRUCTION+1
                ;IN C(3).  C(2) POINTS TO THE BUFFER LOCATION TO BE
                ;EXECUTED.
                ;C(3)-1 INDICATES WHICH LOCATION IN THE BUFFER WAS
                ;EXECUTED. THE
                ;C(3) SHOULD BE ONE GREATER THAN THE BUFFER POINTER IN
                ;C(2).

01214 102520  JSR10:  SUBZL 0,0        ;LOOP ON ERR SWITCH
01215 034114          LDA 3,BUFF       ;C(3)=FIRST BUFFER LOCATION
01216 030421          LDA 2,CJSR10     ;CONSTANT (JSR 1,3)
01217 024115          LDA 1,FIN        ;C(1)=FINAL BUFFER LOCATION
01220 051400          STA 2,0,3        ;FILL THE BUFFER
01221 175400          INC 3,3          ;WITH JSR INSTRUCTIONS
01222 166414          SUB# 3,1,SZR     ;TEST FOR END OF THE BUFFER
01223 000775          JMP .-3          ;MORE WORDS TO STORE

01224 030114  JSR11:  LDA 2,BUFF       ;C(2)=FIRST BUFFER LOCATION
01225 005000          JSR 0,2          ;GO TO THE BUFFER; C(3)=
01226 063077          HALT             ;RETURN ADDR. FAIL TO PC XFER
01227 156014          ADC# 2,3,SZR     ;CHECK NEW C(3) FROM JSR IN
                                       ;BUFF
01230 063077          HALT             ;C(3)SHOULD=C(2)+1. WENT WRONG
01231 156014          ADC# 2,3,SZR
01232 102400          SUB 0,0
01233 113000          ADD 0,2
01234 146414          SUB# 2,1,SZR     ;NEXT BUFF LOC.TEST FOR END
01235 000770          JMP JSR11+1      ;MORE LOCATIONS IN BUFF
01236 101011          MOV# 0,0,SKP     ;SKIP OVER CONSTANTS
01237 005401  CJSR10: JSR 1,3
```

^^^

```
                ;A TEST OF JMP
                ;THIS PROGRAM WILL EXECUTE A JMP TO EVERY LOCATION IN
                ;THE BUFFER.
                ;THE BUFFER CONTAINS A (JSR 1,3) INSTRUCTION IN EACH
                ;LOCATION.
                ;THE VALUE IN C(3) IS SET SUCH THAT THE JSR IN THE
                ;BUFFER WILL
                ;RETURN TO THE MAIN PROGRAM. THE PROGRAM THEN CHECKS
                ;THE PC
                ;PROGRAM COUNTER) STORED BY THE BUFFERS' JSR
                ;INSTRUCTION. THIS
                ;PC SHOULD BE ONE GREATER THAN THE LOCATION JUMPED TO.

01240 126520  JMP10: SUBZL 1,1        ;LOOP ON ERR SWIT
01241 020115         LDA 0,FIN        ;FINAL ADDRESS OF BUFFER
01242 030114         LDA 2,BUFF       ;FIRST ADDRESS OF BUFFER
01243 034413         LDA 3,CJMP11     ;C(3)=A RETURN TO THE PROG
01244 001000         JMP 0,2          ;GO TO A LOCATION IN BUFFER
01245 063077         HALT             ;NO JMP? INDEX+1 IN BUFFER?
01246 156014  JMP11: ADC# 2,3,SZR     ;CHECK PC STORED BY BUFFER JSR
01247 063077         HALT             ;C(2)=JMP ADDRESS,C(3)=JSR PC
01250 156014         ADC# 2,3,SZR
01251 126400         SUB 1,1
01252 133000         ADD 1,2
01253 142414         SUB# 2,0,SZR     ;TEST FOR END OF THE BUFFER
01254 000767         JMP JMP10+3      ;NOT YET END OF BUFFER
01255 101011         MOV# 0,0,SKP     ;SKIP OVER THE CONSTANT
01256 001245  CJMP11: JMP11-1         ;CONSTANT

                .EOT
```

```
                        ;A TEST OF POSITIVE DISPLACEMENT USING THE "LDA"
                        ;INSTRUCTION
                        ;THE BUFFER IS FILLED WITH A ADDRESS PATTERN
                        ;(C(ADDRESS)=ADDRESS).
                        ;C(3) POINTS TO A LOCATION IN THE BUFFER,. A "LDA"
                        ;INSTRUCTION
                        ;WITH A POSITIVE DISPLACEMENT THEN REFFERANCES THE
                        ;BUFFER VIA
                        ;INDEX REGISTER 3.  THE EFFECTIVE ADDRESS IS THE SUM OF
                        ;THE INDEX
                        ;REGISTER VALUE AND THE DISPLACEMENT. C(2)=EFFECTIVE
                        ;ADDRESS
                        ;OBTAINED,C(1)=CORRECT EFFECTIVE ADDRESS,C(3)=INDEXING
                        ;VALUE
                        ;C(POSX2)=FAILING INSTRUCTION--DISPLACEMENT IN BITS
                        ;8-15

01257 034114  POSX:   LDA 3,BUFF      ;FIRST BUFFER LOCATION
01260 024115          LDA 1,FIN       ;LAST BUFFER LOCATION
01261 055400          STA 3,0,3       ;STORE A ADDRESS PATTERN
01262 175400          INC 3,3         ;IN THE BUFFER
01263 136414          SUB# 1,3,SZR    ;C(BUFFER)=ADDRESS OF BUFF
01264 000775          JMP POSX+2      ;EXAMPLE: C(3015)=3015
01265 030426          LDA 2,CLDAPX    ;CONSTANT (LDA 2,0,3) INST
01266 055411          STA 2,POSX2     ;INITIALIZE THE INSTRUCTION

01267 024120  POSX1:  LDA 0,C3        ;ADD 3 TO THE INDEX
                                      ;DISPLACEMENT=-
01270 024407          LDA 1,POSX2     ;MENT OF THE LDA INSTRUCTION
01271 107000          ADD 0,1
01272 044405          STA 1,POSX2
01273 024421          LDA 0,POSFIN    ;END TEST FOR DISPLACEMENT
01274 106032          ADCZ# 0,1,SZC
01275 000420          JMP POSND       ;GO TO NEXT TEST
01276 034114          LDA 3,BUFF      ;IF THIS TEST FAILS PLACE
01277 000000  POSX2:  0               ;(JMP .-1) AFTER THIS
                                      ;INSTRUCTION
01300 024777          LDA 1,.-1       ;C(POSX2)=LDA 2,N,3
01301 020122          LDA 0,C377      ;INSTRUCTION DISPLACEMENT
01302 107400          AND 0,1         ;TO C(1).  THIS DISPLACEMENT
01303 107000          ADD 3,1         ;+INDEX VALUE=EFFECTIVE ADDRESS
01304 146414          SUB# 2,1,SZR    ;C(1)=EFFECTIVE ADDRESS,C(2)=
01305 063077          HALT            ;WORD OBTAINED, C(2) WRONG
01306 020116          LDA 0,FIN200    ;THIS DISPLACEMENT WILL BE
01307 175400          INC 3,3         ;TRIED WITH OTHER LOCATIONS
01310 116414          SUB# 0,3,SZR    ;IN THE BUFFER
01311 000766          JMP POSX2
01312 000755          JMP POSX1       ;NEXT DISPLACEMENT

01313 031400  CLDAPX: LDA 2,0,3      ;CONSTANT FIRST DISPLACEMENT
01314 031577  POSFIN: LDA 2,177,3     ;CONSTANT FINAL "
01315 101010  POSND:  MOV# 0,0       ;REPLACE WITH JMP TO LOOP
```

```
^^^

                ;A TEST OF NEGATIVE DISPLACEMENT USING "LDA"
                ;THE BUFFER IS FILLED WITH A ADDRESS PATTERN
                ;(C(ADDRESS)=ADDRESS). A "LDA" INSTRUCTION
                ;WITH A NEGATIVE DISPLACEMENT THEN REFFERANCES
                ;THE BUFFER VIA INDEX REGISTER 2. THE EFFECTIVE
                ;ADDRESS IS THE VALUE OF THE INDEX REGISTER
                ;MINUS THE DISPLACEMENT VALUE. C(3)=EFFECTIVE
                ;ADDRESS OBTAINED,C(1)=CORECT EFFECTIVE ADDRESS,
                ;C(2)=INDEXING VALUE,C(NEGX2)=FAILING INSTRUCTION
                ;DISPLACEMENT IN BITS 8-15.

01316 034114  NEGX:   LDA 3,BUFF       ;FIRST BUFFER LOCATION
01317 024115          LDA 1,FIN        ;FINAL BUFFER LOCATION
01320 055400          STA 3,0,3        ;FILL THE BUFFER WITH A
01321 175400          INC 3,3          ;ADDRESS PATTERN.FOR
01322 136414          SUB# 1,3,SZR     ;EXAMPLE: C(3417)=3417
01323 000775          JMP NEGX+2       ;C(4150)=4150
01324 030431          LDA 2,NEGFIN     ;INITIALIZE THE LDA INST
01325 050412          STA 2,NEGX2      ;TO (LDA 3,377,2)

01326 020120  NEGX1:  LDA 0,C3         ;SUBTRACT (3) FROM THE LDA'S
01327 024410          LDA 1,NEGX2      ;DISPLACEMENT
01330 106400          SUB 0,1
01331 044406          STA 1,NEGX2      ;PUT IT BACK IN CORE
01332 020424          LDA 0,NEGXX      ;TEST FOR A POSITIVE
01333 106033          ADCZ# 0,1,SNC    ;DISPLACEMENT
01334 000423          JMP NEGND        ;END. GO TO NEXT TEST
01335 000401          JMP NEGX2-1

01336 030117          LDA 2,BUF200     ;FIRST BUFFER LOCATION+200
01337 000000  NEGX2:  0                ;BECOMES: LDA 3,-N,2  ↓↓
01340 024777          LDA 1,.-1        ;GET DISPLACEMENT BITS
01341 020122          LDA 0,C377       ;MASK THE INSTRUCTION PART
01342 107400          AND 0,1
01343 100000          COM 0,0
01344 107000          ADD 0,1          ;C(1)=THE EFFECTIVE ADDRESS
01345 147000          ADD 2,1          ;C(2)=THE INDEXING VALUE
01346 136414          SUB# 1,3,SZR     ;C(3)=THE ADDRESS OBTAINED
01347 063077          HALT             ;INST AT NEGX2 FAILED
01350 020115          LDA 0,FIN        ;USE THIS DISPLACEMENT
01351 151400          INC 2,2          ;WITH OTHER LOCATIONS IN THE
01352 112414          SUB# 0,2,SZR     ;BUFFER
01353 000764          JMP NEGX2        ;NOT BUFFER END
01354 000752          JMP NEGX1        ;END,TRY NEW DISPLACEMENT


01355 035377  NEGFIN: LDA 3,-1,2       ;CONSTANT: DISPLACEMENT=1
01356 035200  NEGXX:  LDA 3,-200,2     ;CONSTANT: DISPLACEMENT OF =177
01357 101010  NEGND:  MOV# 0,0         ;REPLACE WITH JMP TO LOOP TEST
```

^^^

```
                    ;A TEST OF POSITIVE DISPLACEMENT USING THE "JMP".
                    ;THE BUFFER IS FILLED WITH (JSR 0,3) INSTRUCTIONS
                    ;THUS WHEN A JSR IN THE BUFFER IS EXECUTED C(3) WILL
                    ;BE SET TO THE BUFFER LOCATION+1. A "JMP" INSTRUCTION
                    ;WITH A POSITIVE DISPLACEMENT GOES TO THE BUFFER VIA
                    ;INDEX REGISTER 2. THE JSR INSTRUCTION IN THE BUFFER
                    ;WILL RETURN TO THE PROGRAM SAVING ITS LOCATION +1 IN
                    ;C(3). THE EFFECTIVE ADDRESS OF THE "JMP" IS COMPAIRED
                    ;WITH C(3). C(3)-1=LOCATION JUMPED TO. C(2)=INDEX
                    ;VALUE,C(1)=EFFECTIVE ADDRESS,C(POSJ3)=FAILING INST-
                    ;RUCTION,DISPLACEMENT IN BITS 8-15.

01360 034114  POSJ:   LDA 3,BUFF        ;FIRST LOCATION IN BUFFER
01361 024115          LDA 1,FIN         ;FINAL LOCATION IN BUFFER
01362 030435          LDA 2,POSJX       ;FILL THE BUFFER WITH(JSR 0,3)
01363 051400          STA 2,0,3         ;INSTRUCTIONS
01364 175400          INC 3,3
01365 166414          SUB# 3,1,SZR      ;TEST FOR END OF BUFFER
01366 000774          JMP POSJ+2        ;MORE LOCATIONS TO FILL
01367 020427          LDA 0,POSJZ       ;INITIALIZE THE JUMP
01370 040416          STA 0,POSJ3       ;WITH POSITIVE DISPLACEMENT

01371 020121  POSJ1:  LDA 0,C5          ;INCREASE POSITIVE DISPLACE
01372 024414          LDA 1,POSJ3       ;BY FIVE
01373 107000          ADD 0,1
01374 044412          STA 1,POSJ3       ;NEW "JMP" INSTRUCTION
01375 020423          LDA 0,POSJF       ;ANY MORE + DISPLACEMENTS
01376 106032          ADCZ# 0,1,SZC     ;TO GO
01377 000423          JMP POSJND        ;NO GO TO NEXT TEST

01400 030114  POSJ2:  LDA 2,BUFF        ;FIRST LOCATION IN BUFFER
01401 020116          LDA 0,FIN200      ;FINAL LOCATION IN BUFFER
01402 151400          INC 2,2           ;POINT TO NEXT BUFFER
01403 142415          SUB# 2,0,SNR      ;LOCATION. IS IT THE LAST?
01404 000765          JMP POSJ1         ;YES TRY NEW DISPLACEMENT
01405 034414          LDA 3,POSJY       ;SETUP C(3) FOR BUFFER RET
01406 000000  POSJ3:  0                 ;"THE" JMP INSTRUCTION
01407 024777          LDA 1,.-1
01410 020122          LDA 0,C377        ;CALCULATE THE EFFECTIVE
01411 107400          AND 0,1           ;ADDRESS OF THIS JUMP
01412 147000          ADD 2,1           ;C(INDEX)+DISPLACEMENT
01413 136014          ADC# 1,3,SZR      ;C(1)=EFFECTIVE ADDRESS,C(3)
01414 063077          HALT              ;JSR PC,C(2)=INDEX VALUE
01415 000764          JMP POSJ2+1       ;GO AGAIN AT NEXT LOC

01416 001000  POSJZ:  JMP 0,2           ;CONSTANTS
01417 005400  POSJX:  JSR 0,3           ;"
01420 001177  POSJF:  JMP 177,2
01421 001407  POSJY:  POSJ3+1
01422 101010  POSJND: MOV# 0,0         ;REPLACE WITH JMP TO LOOP
```

^^^

```
                        ;A TEST OF NEGATIVE DISPLACEMENT USING "JSR"
                        ;THE BUFFER IS FILLED WITH (JSR 0,3) INSTRUCTIONS.
                        ;THE BUFFER IS ENTERED WITH A JSR VIA C(3) WITH A
                        ;NEGATIVE DISPLACEMENT IN BITS 8-15. EFFECTIVE
                        ;ADDRESS CALCULATION SHOULD SUBTRACT THE DIS-
                        ;PLACEMENT FROM THE VALUE IN INDEX REGISTER 3.
                        ;THE JSR IN THE BUFFER WILL RETURN TO THE PROGRAM
                        ;SAVING ITS LOCATION +1 IN C(3). COMPARISON OF
                        ;CALCULATED EFFECTIVE ADDRESS AND C(3) DETERMIN
                        ;IF THE BUFFER WAS ENTERED CORRECTLY.

01423 030432  NEGJ:     LDA 2,NEGJY       ;C(2)=CONSTANT (JSR -1,2)
01424 050415            STA 2,NEGJ3       ;STORE "THE" JSR

01425 020120  NEGJ1:    LDA 0,C3          ;DECREMENT THE VALUE OF
01426 024413            LDA 1,NEGJ3       ;THE DISPLACEMENT BY 3.
01427 106400            SUB 0,1           ;EACH DISPLACEMENT IS TEST-
01430 044411            STA 1,NEGJ3       ;ED WITH THE BUFFER
01431 020423            LDA 0,NEGJX
01432 106052            ADC0# 0,1,SZC     ;DISPLACEMENT END TEST
01433 000423            JMP NEGJD         ;GO TO NEXT TEST

01434 030117  NEGJ2:    LDA 2,BUF200      ;FIRST ADDRESS+200 IN BUFFER
01435 020115            LDA 0,FIN         ;THIS ROUTINE SETS CARRY!!!
01436 151440            INC0 2,2          ;ITS TESTED LATER! INC BUFF
01437 142435            SUBZ# 2,0,SNR     ;ADDRESS AND TEST FOR END
01440 000765            JMP NEGJ1         ;END,GET NEW "JSR"

01441 000000  NEGJ3:    0                 ;GO TO THE BUFFER HERE!
01442 024777            LDA 1,.-1         ;OBTAIN THE EFFECTIVE
01443 020122            LDA 0,C377        ;ADDRESS IN C(1)
01444 107403            AND 0,1,SNC       ;THE CARRY SHOULD NOT HAVE
01445 063077            HALT              ;CHANGED! SEE ABOVE
01446 100000            COM 0,0
01447 107000            ADD 0,1
01450 147000            ADD 2,1           ;CHECK EFFECTIVE ADDRESS
01451 136014            ADC# 1,3,SZR      ;CHECK EFFECTIVE ADDRESS
01452 063077            HALT              ;WITH PC OF BUFFER JSR IN
01453 000762            JMP NEGJ2+1       ;C(3),C(2)=INDEX VALUE

01454 005200  NEGJX:    JSR -200,2        ;CONSTANT
01455 005377  NEGJY:    JSR -1,2          ;CONSTANT
01456 101010  NEGJD:    MOV# 0,0          ;REPLACE WITH JMP TO LOOP
```

^^^

```
                        ;TEST THAT PC WILL COUNT THROUGH THE BUFFER.

01457 034114  PINC:   LDA 3,BUFF
01460 024115          LDA 1,FIN
01461 020415          LDA 0,CINC        ;FILL THE BUFFER
01462 041400  PINC1:  STA 0,0,3         ;WITH INC INSTRUCTIONS
01463 175400          INC 3,3
01464 136414          SUB# 1,3,SZR
01465 000775          JMP PINC1
01466 020411          LDA 0,CPINC
01467 041400          STA 0,0,3         ;PUT JMP IN LAST LOC
01470 030410  PINC2:  LDA 2,PINCR       ;C(2)=RETURN TO PROG
01471 034114          LDA 3,BUFF
01472 001400          JMP 0,3           ;GO TO BUFFER

01473 136414  PINC3:  SUB# 1,3,SZR      ;BUFER INST SHOULD COUNT
01474 063077          HALT              ;C(1)=CORRECT
01475 000404          JMP .+4           ;C(3)=BUFF+INC RESULT
01476 175400  CINC:   INC 3,3
01477 001000  CPINC:  JMP 0,2
01500 001473  PINCR:  PINC3
01501 101000          MOV 0,0           ;CHANGE TO LOOP TEST

                      .EOT
```

```
                ;A TEST OF ISZ ABILITY TO NOT SKIP
                ;THE C(TEM) IS SET TO 0.INCREMENTING TO 000001 SHOULD
                ;NOT CAUSE A SKIP. PRESSING CONTINUE AFTER A ERROR WILL
                ;CAUSE THE PROGRAM TO LOOP ON ERROR UNTILL RESTARTED.

01502 102520    ISZ0:   SUBZL 0,0       ;C(0)=1,LOOP ON ERROR SWITCH
01503 024122            LDA 1,C377      ;NUMBER OF ITERATIONS
01504 152000            ADC 2,2         ;C(2) NOT USED THIS TEST
01505 176400            SUB 3,3         ;THE VALUE STORED (0)
01506 054107    ISZ1:   STA 3,TEM       ;STORE FOR INCREMENT
01507 060377            NIOP CPU        ;SYNC AT 7-A74
01510 010107            ISZ TEM         ;INC A ZEROS WORD
01511 000404            JMP .+4         ;JMP PAST THE ERROR
01512 020107            LDA 0,TEM       ;ISZ SKIPPED C(0)=RESULT
01513 063077            HALT            ;OF ISZ. CORRECT VALUE= +1
01514 102400            SUB 0,0         ;LOOP ON ERR SWITCH
01515 106404            SUB 0,1,SZR     ;ITERATE THE PROGRAM
01516 000770            JMP ISZ1
01517 150014            COM# 2,2,SZR    ;C(2) MODIFIED!! CHECK ISZ
01520 063077            HALT            ;REPLACE TO LOOP TEST

                ;A TEST OF ISZ ABILITY TO NOT SKIP
                ;THE C(TEM) IS SET TO -2 (177776).INCREMENTION TO -1
                ;SHOULD NOT CAUSE A SKIP. PRESSING CONTINUE AFTER A
                ;ERROR WILL CAUSE THE PROGRAM TO LOOP ON ERROR UNTILL
                ;RESTARTED.

01521 102520    ISZ2:   SUBZL 0,0       ;C(0)=1,LOOP ON ERROR SWITCH
01522 024122            LDA 1,C377      ;C(1)=NUMBER OF ITERATIONS
01523 152400            SUB 2,2         ;C(2) NOT USED THIS TEST
01524 176120            ADCZL 3,3       ;C(3)=-2 (177776)
01525 054107    ISZ3:   STA 3,TEM       ;STORE AND THEN INCREMENT IT
01526 060377            NIOP CPU        ;SYNC AT 7-A74
01527 010107            ISZ TEM         ;ISZ SHOULD NOT SKIP
01530 000404            JMP .+4         ;JMP PAST THE ERROR
01531 020107            LDA 0,TEM       ;C(0)=ISZ RESULT
01532 063077            HALT            ;IT SHOULD BE 177777
01533 102401            SUB 0,0,SKP     ;ISZ LOGIC FAIL
01534 106404            SUB 0,1,SZR     ;ITERATION COUNTER
01535 000770            JMP ISZ3
01536 151014            MOV# 2,2,SZR    ;C(2) MODIFIED !!
01537 063077            HALT            ;CHANGE MOV TO LOOP TEST
```

^^^

```
                        ;A TEST OF ISZ ABILITY TO SKIP
                        ;THE C(TEM) IS SET TO -1 (177777). INCREMENTIOG TEM
                        ;TO 0 SHOULD PRODUCE A ISZ SKIP.

01540 102520  ISZ4:     SUBZL 0,0           ;C(0)=1,LOOP ON ERROR SWITCH
01541 126620            SUBZR 1,1           ;C(1)=100000,ITERATION COUNT
01542 152000            ADC 2,2             ;C(2)=177777
01543 050107  ISZ5:     STA 2,TEM           ;STORE THE (-1)
01544 060377            NIOP CPU            ;SYNC AT 7-A74
01545 010107            ISZ TEM             ;INC IT SHOULD SKIP!
01546 000404            JMP ISZ6            ;FAIL TO SKIP,ISZ FAIL
01547 107004            ADD 0,1,SZR         ;ITERATION COUNTER
01550 000773            JMP ISZ5            ;CONTINUE TEST
01551 000405            JMP ISZ7            ;END OF THIS TEST
01552 020107  ISZ6:     LDA 0,TEM           ;C(0)=RESULT OF FAIL ISZ
01553 063077            HALT                ;CORRECT VALUE=0
01554 102400            SUB 0,0             ;CLEAR C(0) TO LOOP
01555 000766            JMP ISZ5            ;ON ERROR FOREVER
01556 101234  ISZ7:     MOVZR# 0,0,SZR      ;REPLACE TO LOOP TEST
01557 063077            HALT                ;C(0) SHOULD BE +1

                        ;THE RESULT OF A ISZ INSTRUCTION IS CHECKED WITH A ADD.
                        ;ADD IS PRESUMED TO WORK.THE ISZ INCREMENT LOGIC,SKIP
                        ;AND NO SKIP IS CHECKED.PRESSING CONTINUE AFTER A ERROR
                        ;WILL CAUSE THE PROGRAM TO LOOP UNTILL RESTARTED.

01560 152520  ISZ10:    SUBZL 2,2           ;C(2)=1
01561 102620            SUBZR 0,0           ;C(0)=100000
01562 143005            ADD 2,0,SNR         ;UPDATE THE COUNT BY 1

01563 000422            JMP ISZ13           ;ISZ FAIL TO SKIP ON 177777
01564 040107            STA 0,TEM           ;STORE THE VALUE FOR ISZ
01565 060377            NIOP CPU            ;SYNC A 7-A74
01566 010107            ISZ TEM             ;"THE" ISZ
01567 101011            MOV# 0,0,SKP        ;ISZ SHOULD SKIP AT END OF TEST
01570 000410            JMP ISZ12           ;ONLY
01571 024107            LDA 1,TEM           ;DOES VALUE OF TEM=COUNT+1
01572 106015            ADC# 0,1,SNR        ;IN C(0)
01573 000767            JMP ISZ10+2         ;YES KEEP CHECKING

01574 115400  ISZ11:    INC 0,3             ;C(1)=ISZ COUNT,C(3)=CORRECT
01575 063077            HALT                ;ISZ INCREMENT LOGIC FAILED
01576 152400            SUB 2,2             ;CLEAR C(2) TO LOOP ON ERROR
01577 000763            JMP ISZ10+2
01600 115405  ISZ12:    INC 0,3,SNR         ;ISZ SKIPPED WAS THAT OK?
01601 000410            JMP ISZ14           ;YES. EXIT THIS TEST
01602 024107            LDA 1,TEM           ;C(1)=ISZ COUNT,C(3)=INC COUNT
01603 063077            HALT                ;C(1)SHOULD = C(3),SKIP OR NOT
01604 000772            JMP ISZ12-2         ;CLEAR COUNT AND LOOP ERROR
01605 102000  ISZ13:    ADC 0,0             ;ISZ FAIL TO SKIP ON
01606 024107            LDA 1,TEM           ;C(TEM)=-1,C(1)=ISZ COUNT
01607 063077            HALT                ;LOOP ON ERROR
01610 000766            JMP ISZ12-2
01611 101010  ISZ14:    MOV# 0,0            ;CHANGE TO LOOP TEST
```

```
^^^

                ;A TEST OF ISZ
                ;FIVE CONSECUTIVE ISZ INSTRUCTIONS TO LOCATION"TEM" ARE
                ;EXECUTED.THE VALUE IN C(TEM) SHOULD EQUAL C(3)
                ;THE NUMBERS ARE ARRANGED SUCH THAT NO ISZ INSTRUCTION
                ;SHOULD SKIP. SHOULD THE TEST FAIL PRESSING CONTINUE
                ;WILL PLACE THE PROGRAM IN A FAILING LOOP UNTILL
                ;RESTARTED.

01612 030121    ISZ20:  LDA 2,C5        ;C(2)=ERROR SWITCH
01613 102400            SUB 0,0         ;START WITH C(0)=0
01614 034121            LDA 3,C5        ;C(3)=EXPECTED RESULT OF
01615 117022            ADDZ 0,3,SZC    ;ISZ. ALSO TEST FOR END
01616 000417            JMP ISZ22       ;GO TO NEXT TEST
01617 040107    ISZ21:  STA 0,TEM       ;STORE THE VALUE TO BE INC
01620 060377            NIOP CPU        ;SYNC AT 7-A74
01621 010107            ISZ TEM         ;DO IT 5 TIMES
01622 010107            ISZ TEM
01623 010107            ISZ TEM
01624 010107            ISZ TEM
01625 010107            ISZ TEM
01626 024107            LDA 1,TEM       ;C(1)=RESULT OF ISZ*5
01627 136414            SUB# 1,3,SZR    ;C(3)=CORRECT RESULT
01630 063077            HALT            ;ISZ FAILED
01631 136414            SUB# 1,3,SZR    ;IF A ERROR OCCURED
01632 152400            SUB 2,2         ;CLEAR C(0) TO LOOP ERR
01633 143000            ADD 2,0         ;UPDATE FOR NEXT GO
01634 000760            JMP ISZ20+2     ;ROUND
01635 101010    ISZ22:  MOV# 0,0        ;CHANGE TO LOOP TEST

                ;EACH WORD IN THE BUFFER IS SET TO -1 (177777).
                ;ISZ THEN INCREMENTS IT TO ZERO AND SKIPS. THE PROGRAM
                ;WILL LOOP ON ERROR IF CONTINUE IS PRESSED

01636 102520    ISZ30:  SUBZL 0,0       ;C(0)=1,THE LOOP ON ERR SWIT
01637 024115            LDA 1,FIN       ;C(1)=FINAL BUFFER ADDRESS
01640 030114            LDA 2,BUFF      ;C(2)=FIRST BUFFER ADDRESS
01641 176000            ADC 3,3         ;C(3)=177777 (-1)
01642 055000    ISZ31:  STA 3,0,2       ;STORE THE (-1) IN BUFFER
01643 060377            NIOP CPU        ;SYNC AT 7-A74
01644 011000            ISZ 0,2         ;INCREMENTION SHOULD SKIP
01645 000412            JMP ISZ32+2     ;ISZ FAIL TO SKIP
01646 035000            LDA 3,0,2       ;DOES BUFFER=0?
01647 175014            MOV# 3,3,SZR    ;IF NOT ISZ FAIL TO COUNT
01650 000405            JMP ISZ32       ;CORRECT , BUT SKIPPED
01651 113000            ADD 0,2         ;INC TO NEXT BUFFER LOC
01652 132414            SUB# 1,2,SZR    ;AND TEST FOR END
01653 000766            JMP ISZ31-1
01554 000407            JMP ISZ33       ;END GO NEXT TEST
01655 063077    ISZ32:  HALT            ;ISZ SKIPPED BUT STORED NG
01656 000403            JMP .+3         ;C(3)=ISZ RESULT,C(2)=OK
01657 035000            LDA 3,0,2       ;C(2)=ADDRESS OF ERR
01660 063077            HALT            ;ISZ FAIL TO SKIP. C(3)=
01661 102400            SUB 0,0         ;ISZ RESULT.C(2)=EFF ADDRESS
01662 000757            JMP ISZ31-1     ;LOOP ON THE ERROR
01663 101010    ISZ33:  MOV# 0,0        ;CHANGE TO LOOP TEST
```

^^^

```
                ;ANOTHER TEST OF ISZ
                ;THIS TEST WILL ADD ONE TO EACH LOCATION OF A BUFFER
                ;THAT IS PRESET TO ONES. THE LAST LOCATION IN THE
                ;BUFFER IS SET TO CONTAIN ZEROS, THUS PREVENTING
                ;A ISZ SKIP ON THAT LOCATION. A ISZ INSTRUCTION WILL
                ;REFFERANCE THE BUFFER VIA INDEX REGISTER 2.IF THE
                ;ISZ INSTRUCTION SKIPS AC2 IS INCREMENTED AND THE
                ;SEQUENCE REPEATS. WHEN THE ISZ FAILS TO SKIP THE
                ;BUFFER POINTER IS CHECKED. IF IT IS EQUAL TO THE END
                ;OF THE BU
                ;BUFFER THE TEST IS SUCESSFUL. IF HOWEVER IT IS NOT
                ;THE BUFFER END THE PROGRAM WILL HALT. IF A ERROR
                ;OCCURES THE ENTIRE TEST WILL BE REPEATED UNTILL THE
                ;PROGRAM IS RESTARTED.

01664 126400    ISZ40:  SUB 1,1         ;LOOP ON ERR SWITCH
01665 102000            ADC 0,0
01666 034115            LDA 3,FIN       ;FINAL BUFFER ADDRESS
01667 030114            LDA 2,BUFF      ;FIRST BUFFER ADDRESS
01670 041000            STA 0,0,2       ;FILL THE BUFFER WITH
01671 151400            INC 2,2         ;(-1)
01672 156414            SUB# 2,3,SZR    ;EXCEPT THE LAST
01673 000775            JMP .-3         ;WHICH IS FILLED WITH
01674 045400            STA 1,0,3       ;ZEROS

01675 030114    ISZ41:  LDA 2,BUFF      ;C(2) POINTS TO BUFFER
01676 060377            NIOP CPU        ;SYNC AT 7-A74
01677 011000            ISZ 0,2         ;ISZ SHOULD SKIP EVERY
01700 000403            JMP ISZ42       ;TIME EXCEPT LAST
01701 151400            INC 2,2         ;POINT NEXT LOCATION
01702 000774            JMP ISZ41+1

01703 021000    ISZ42:  LDA 0,0,2       ;C(0)=RESULT OF ISZ
01704 156414            SUB# 2,3,SZR    ;SHOULD BE A 0
01705 126001            ADC 1,1,SKP     ;C(1)=LOOP ON ERR SWIT
01706 156414            SUB# 2,3,SZR    ;C(2)=EFF ADDRESS OF ISZ
01707 063077            HALT            ;OPERAND.
01710 125004            MOV 1,1,SZR     ;IF C(1) NOT 0 PROG
01711 000754            JMP ISZ40+1     ;WILL LOOP
```

^^^

```
                ;A TEST OF DSZ ABILITY TO NOT SKIP
                ;THE C(TEM) IS SET TO (-1). DECREMENTING SHOULD
                ;PRODUCE -2 AND NOT SKIP. PRESSING CONTINUE AFTER A
                ;ERROR WILL CAUSE THE PROGRAM TO LOOP ON ERROR
                ;UNTILL RESTARTED.

01712 102520  DSZ0:   SUBZL 0,0         ;C(0)=1,LOOP ON ERR SWITCH
01713 024122          LDA 1,C377
01714 152000          ADC 2,2           ;C(2) NOT USED THIS TEST
01715 176000          ADC 3,3           ;C(3)=-1

01716 054107  DSZ1:   STA 3,TEM         ;STORE THE VALUE
01717 060377          NIOP CPU          ;SYNC AT 7-A74
01720 014107          DSZ TEM           ;DECREMENT  C(TEM)
01721 000404          JMP .+4           ;IT SHOULD NOT SKIP
01722 020107          LDA 0,TEM         ;IT SKIPPED. C(0)=DSZ RESULT
01723 063077          HALT              ;CORRECT VALUE =-2
01724 102401          SUB 0,0,SKP       ;CLEAR C(0) TO LOOP ERR
01725 106404          SUB 0,1,SZR       ;CHANGE AND TEST ITERATION
01726 000770          JMP DSZ1          ;COUNTER. ANOTHER PASS
01727 150014          COM# 2,2,SZR      ;C(2)MODIFIED !
01730 063077          HALT

                ;A TEST OF DSZ ABILITY TO SKIP
                ;C(TEM) IS SET TO 0. DECREMENTING C(TEM) TO 0
                ;SHOULD PRODUCE A DSZ SKIP. PRESSING CONTINUE
                ;AFTER A ERROR WILL CAUSE TOHE PROGRAM TO LOOP
                ;UNTILL RESTARTED.

01731 102520  DSZ2:   SUBZL 0,0         ;C(0)=1,LOOP ON ERR SWITCH
01732 126620          SUBZR 1,1         ;C(1)=100000,ITERATIONS
01733 152520          SUBZL 2,2         ;C(2)=1
01734 050107  DSZ3:   STA 2,TEM         ;STORE THE 0
01735 060377          NIOP CPU          ;SYNC AT 7-A74
01736 014107          DSZ TEM           ;DECREMENT. IT SHOULD SKIP
01737 000404          JMP DSZ4          ;DSZ FAIL TO SKIP
01740 107004          ADD 0,1,SZR       ;TEST ITERATION COUNT
01741 000773          JMP DSZ3          ;CONTINUE ON
01742 000405          JMP DSZ5          ;NEXT TEST
01743 020107  DSZ4:   LDA 0,TEM         ;C(0)=RESULT OF FAIL DSZ
01744 063077          HALT              ;CORRECT COUNT=0
01745 102400          SUB 0,0           ;CLEAR C(0) TO LOOP
01746 000766          JMP DSZ3
01747 101234  DSZ5:   MOVZR# 0,0,SZR    ;C(0) SHOULD BE +1
01750 063077          HALT              ;CHANGE MOVZR TO LOOP TEST
```

^^^

```
                        ;A TEST OF DSZ
                        ;THE RESULT OF A DSZ INSTRUCTION IS CHECKED
                        ;WITH THE RESULT OF A ADD INSTRUCTION.  THE ADD
                        ;IS PRESUMED TO WORK BECAUSE OF PREVIOUS TESTING.
                        ;THE DECREMENT LOGIC OF THE DSZ, ITS ABILITY
                        ;TO SKIP, AND NOT SKIP IS CHECKED.  SHOULD THE
                        ;DSZ FAIL, THE PROGRAM WILL HALT.  PRESSING
                        ;CONTINUE WILL PLACE THE PROGRAM IN A FAILING LOOP
                        ;FOR SCOPING PURPOSES (YOU MUST REMOVE THE HALT).  THE
                        ;PROGRAM
                        ;MUST BE RESTARTED TO GET OUT OF THE FAILING LOOP

01751 152000  DSZ10:    ADC 2,2             ;C(2)=-1
01752 176000            ADC 3,3             ;C(3)=-1
01753 102620            SUBZR 0,0
01754 101400            INC 0,0             ;C(0)=100001
01755 143025            ADDZ 2,0,SNR        ;TEST FOR PASSING THROUGH 0
01756 000424            JMP DSZ13           ;DSZ FAILED TO SKIP
01757 040107            STA 0,TEM           ;STORE THE VALUE FOR DSZ
01760 060377            NIOP CPU            ;SYNC AT 7-A74
01761 014107            DSZ TEM             ;T. DECREMENT. THE DSZ SKIPS
01762 176001            ADC 3,3,SKP         ;ON EXIT. SET C(3)=-1
01763 000411            JMP DSZ12           ;DSZ SKIPPED.
01764 024107            LDA 1,TEM           ;COUNT OBTAINED FROM DSZ
01765 122015            ADC# 1,0,SNR        ;IS COUNT CORRECT?
01766 000767            JMP DSZ10+4         ;YES CONTINUE ON

01767 117000  DSZ11:    ADD 0,3             ;C(3)=CORRECT COUNT
01770 063077            HALT                ;C(1)=DSZ COUNT
01771 152400            SUB 2,2             ;CLEARING AC2 WILL CAUSE
01772 176000            ADC 3,3
01773 000762            JMP DSZ10+4         ;PROGRAM TO LOOP ON ERR

01774 101235  DSZ12:    MOVZR# 0,0,SNR      ;DSZ SKIPPED,WAS THAT OK?
01775 000412            JMP DSZ14           ;YES GO TO NEXT TEST
01776 024107            LDA 1,TEM           ;C(1)=DSZ COUNT CAUSING
01777 117000            ADD 0,3             ;THE SKIP. C(3)=CORRECT
02000 063077            HALT                ;DSZ SKIP LOGIC FAIL
02001 000770            JMP DSZ11+2         ;LOOP ON THE ERROR

02002 024107  DSZ13:    LDA 1,TEM           ;DSZ FAILED TO SKIP
02003 176400            SUB 3,3
02004 102400            SUB 0,0             ;FROM 0,C(1)=DSZ COUNT
02005 063077            HALT                ;C(3)=CORRECT COUNT
02006 000763            JMP DSZ11+2         ;LOOP ON ERR

02007 101010  DSZ14:    MOV# 0,0            ;CHANGE TO LOOP TEST

02010 000420            JMP 2030
      002030            .LOC 2030
```

```
AAA

                ;A TEST OF DSZ
                ;FIVE CONSECUTIVE DSZ INSTRUCTIONS TO LOCATION TEM
                ;ARE EXECUTED. THE VALUE IN TEM SHOULD EQUAL C(3)
                ;WHICH IS 5 LESS THAN THE INITIAL VALUE STORED IN
                ;C(TEM)
                ;THE NUMBERS ARE ARANGED SUCH THAT NO DSK INSTRUCTION
                ;SHOULD SKIP.

02030 030121  DSZ20: LDA 2,C5         ;C(2)=ERROR LOOP SWITCH
02031 102620         SUBZR 0,0        ;START AT 100000
02032 034110         LDA 3,CM5        ;C(3)=EXPECTED DSZ RESULT
02033 117000         ADD 0,3
02034 040107  DSZ21: STA 0,TEM        ;STORE THE VALUE FOR DEC
02035 060377         NIOP CPU         ;SYNC AT 7-A74
02036 014107         DSZ TEM          ;DECREMENT C(TEM)
02037 014107         DSZ TEM
02040 014107         DSZ TEM
02041 014107         DSZ TEM
02042 014107         DSZ TEM
02043 024107         LDA 1,TEM        ;C(1)=RESULT OF DECREMENT
02044 136414         SUB# 1,3,SZR     ;C(3)=CORRECT COUNT
02045 063077         HALT             ;DSZ FAILED
02046 136414         SUB# 1,3,SZR
02047 152400         SUB 2,2          ;CLEAT C(2) TO LOOP
02050 143023         ADDZ 2,0,SNC     ;UPDATE FOR NEXT
02051 000761         JMP DSZ20+2      ;GO ROUND
02052 101010  DSZ22: MOV# 0,0         ;CHANGE TO LOOP TEST

                ;A TEST OF DSZ
                ;EACH WORD IN THE BUFFER IS SET TO +1. DSZ THEN
                ;DECREMENTS IT TO (+1) AND SKIPPES. IF A ERROR
                ;OCCURES THE PROGRAM WILL CYCLE CONTINUSELY
                ;UNTILL IT IS RESTARTED.

02053 102520  DSZ30: SUBZL 0,0        ;C(0)=1,LOOP ON ERR SWITCH
02054 024115         LDA 1,FIN        ;C(1)=FINAL BUFFER ADDRESS
02055 030114         LDA 2,BUFF       ;C(2)=FIRST BUFFER ADDRESS
02056 176520         SUBZL 3,3        ;C(3)=+1
02057 055000  DSZ31: STA 3,0,2        ;STORE WORD OF 0 IN BUFFER
02060 060377         NIOP CPU         ;SYNC AT 7-A74
02061 015000         DSZ 0,2          ;DECREMENTION SHOULD SKIP
02062 000412         JMP DSZ32+2      ;DSZ FAIL TO SKIP
02063 035000         LDA 3,0,2        ;DSZ SKIPPED OK,IS MEM
02064 175014         MOV# 3,3,SZR     ;WORD 0?
02065 000405         JMP DSZ32        ;NO! ERROR
02066 113000         ADD 0,2          ;ADVANCE TO NEXT LOCATION
02067 132414         SUB# 1,2,SZR
02070 000766         JMP DSZ31-1      ;MORE LOCATIONS IN BUFFER
02071 000407         JMP DSZ33        ;GO TO NEXT TEST
02072 063077  DSZ32: HALT             ;DSZ SKIPPED BUT STORE NG
02073 000403         JMP .+3          ;C(2)=EFF ADDRESS,C(3)=RESULT
02074 035000         LDA 3,0,2        ;OF THE DSZ
02075 063077         HALT             ;DSZ FAIL TO SKIP
02076 102400         SUB 0,0          ;C(2)=EFFECTIVE ADDRESS
02077 000757         JMP DSZ31-1      ;LOOP THIS TEST
02100 101010  DSZ33: MOV# 0,0         ;CHANGE TO LOOP TEST
```

```
^^^

                        ;ANOTHER TEST OF DSZ
                        ;THIS TEST WILL SUBTRACT ONE FROM EACH LOCATION OF A
                        ;BUFFER SET TO +1.THE TEST HAS BEEN WRITTEN TO GIVE
                        ;A HIGH REP RATE FOR THE DSZ INSTRUCTION. PRESSING
                        ;CONTINUE AFTER A ERROR WILL CAUSE THE TEST TO BE
                        ;ITERATED UNTILL RESTARTED.

02101 126000  DSZ40:  ADC 1,1          ;C(1)=LOOP ON ERROR SWITCH
02102 102520          SUBZL 0,0        ;C(0)=+1
02103 034115          LDA 3,FIN
02104 030114          LDA 2,BUFF
02105 041000          STA 0,0,2        ;SET ALL LOCATIONS IN
02106 151400          INC 2,2          ;THE BUFFER TO ZERO
02107 156414          SUB# 2,3,SZR     ;EXCEPT THE LAST LOCATION
02110 000775          JMP .-3
02111 051400          STA 2,0,3        ;DSZ SKIPS ALL BUT LAST

02112 102000  DSZ41:  ADC 0,0          ;C(0)=-1,FOR ERROR CHECK
02113 034111          LDA 3,C4         ;C(3)=4,BUFFER INCREMENT
02114 030114          LDA 2,BUFF       ;C(2) POINTS TO THE BUFFER

02115 173000  DSZ42:  ADD 3,2          ;INC BUFFER POINTER BY 4
02116 060377          NIOP CPU         ;SYNC 7-A74
02117 015374          DSZ -4,2         ;DECREMENT THE BUFFER
02120 000410          JMP DSZ43        ;VIA DSZ. DSZ SHOULD SKIP
02121 015375          DSZ -3,2         ;UNLESS END OF BUFFER
02122 000407          JMP DSZ43+1
02123 015376          DSZ -2,2
02124 000406          JMP DSZ43+2
02125 015377          DSZ -1,2
02126 000405          JMP DSZ43+3
02127 000766          JMP DSZ42        ;NOT YET BUFFERS END

02130 113000  DSZ43:  ADD 0,2          ;C(2) WILL BE SET
02131 113000          ADD 0,2          ;TO THE EFFECTIVE ADDRESS
02132 113000          ADD 0,2          ;OF THE ENTERING DSZ
02133 113000          ADD 0,2          ;INSTRUCTION
02134 034115          LDA 3,FIN        ;DID DSZ ENTER AT END OF
02135 021000          LDA 0,0,2        ;BUFFER OR DID IT FAIL?
02136 156414          SUB# 2,3,SZR     ;C(3)=END OF BUFFER
02137 126400          SUB 1,1          ;C(2)=EFFECTIVE ADDRESS
02140 156414          SUB# 2,3,SZR     ;C(0)=RESULT OF DSZ INST
02141 063077          HALT             ;A DSZ FAILED TO SKIP
02142 124014          COM# 1,1,SZR     ;IF C(1)=0 LOOP ON ERROR
02143 000737          JMP DSZ40+1

                      .EOT
```

```
                ;A TEST OF INDIRECT ADDRESSING
                ;THE C(TEM) IS SET TO POINT TO A BUFFER. A LDA
                ;INSTRUCTION THEN REFFERANCES THE BUFFER INDIRECTLY.
                ;THE C(TEM) IS CHECKED TO INSURE THAT IT HAS NOT
                ;CHANGED. PRESSING CONTINUE AFTER A ERROR WILL CAUSE
                ;THE PROGRAM TO LOOP ON THE FAILING CONDITIONS
                ;UNTILL RESTARTED.
02144 102520    DEF0:   SUBZL 0,0       ;C(0)=1,LOOP ON ERR SWITCH
02145 030114            LDA 2,BUFF      ;FIRST ADDRESS IN BUFFER
02146 034115            LDA 3,FIN       ;FINAL ADDRESS IN BUFFER
02147 050107            STA 2,TEM       ;C(2) STORED IN C(TEM)
02150 060377            NIOP CPU        ;SYNC AT 7-A74
02151 026107            LDA 1,@TEM      ;REFFERANCE REGISTER TEM
02152 024107            LDA 1,TEM       ;HAS C(TEM) CHANGED
02153 132414            SUB# 1,2,SZR    ;IF+OR- ONE PERHAPS AUTO IDX
02154 063077            HALT            ;YES,C(1)=VALUE OF TEM
02155 132414            SUB# 1,2,SZR    ;C(2)=VALUE STORED IN TEM
02156 102400            SUB 0,0         ;CLEAR C(0) TO LOOP
02157 113000            ADD 0,2         ;INC BUFFER POINTER
02160 156414            SUB# 2,3,SZR    ;TEST FOR END OF BUFF
02161 000766            JMP DEF0+3
02162 101010            MOV# 0,0        ;CHANGE TO LOOP TEST

                ;A TEST OF INDIRECT ADDRESSING
                ;A ADDRESS PATTERN (C(ADDRESS)=ADDRESS) IS STORED IN
                ;THE BUFFER VIA INDEX REGISTER 3. A LDA INSTRUCTION
                ;THEN
                ;REFFERANCES EACH LOCATION IN THE BUFFER INDIRECTLY.
                ;BECAUSE THE BUFFER CONTAINS A ADDRESS PATTERN THE DATA
                ;OBTAINED REPRESENTS THE ADDRESS OBTAINED. PRESSING
                ;CONTINUE AFTER A ERROR CAUSES THE PROGRAM TO LOOP ON
                ;THE
                ;ERROR UNTILL RESTARTED.

02163 102520    DEF10:  SUBZL 0,0       ;C(0)=1,LOOP ON ERR SWITCH
02164 034114            LDA 3,BUFF      ;FIRST BUFFER LOCATION
02165 030115            LDA 2,FIN       ;FINAL BUFFER LOCATION
02166 055400            STA 3,0,3       ;STORE THE ADDRESS
02167 175400            INC 3,3         ;PATTERN IN THE BUFFER
02170 156414            SUB# 2,3,SZR    ;EXAMPLE: C(4105)=4105
02171 000775            JMP .-3
02172 034114            LDA 3,BUFF      ;ADDRESS PATTERN IS STORED
02173 054107    DEF11:  STA 3,TEM       ;PRESET C(TEM) FOR INDIRECT
02174 060377            NIOP CPU        ;SYNC AT 7-A74
02175 026107            LDA 1,@TEM      ;GO INDITECT TO GET DATA
02176 136414            SUB# 1,3,SZR    ;C(1)=DATA/ADDRESS OBTAINED
02177 063077            HALT            ;C(3)=CORRECT DATA/ADDRESS
02200 136414            SUB# 1,3,SZR    ;IF IT FAILS
02201 102400            SUB 0,0         ;CLEAR 0 TO LOOP ON ERROR
02202 117000            ADD 0,3         ;ADVANCE TO NEXT ADDRESS
02203 156414            SUB# 2,3,SZR    ;TEST FOR END OF BUFFER
02204 000767            JMP DEF11
02205 101010            MOV# 0,0        ;CHANGE TO LOOP TEST
```

```
                        ;A TEST OF INDIRECT ADDRESSING
                        ;WE DO NOT YET KNOW THAT INDIRECT ADDRESSING WORKS
                        ;THIS TEST INSURES THAT THE STA INST STORES INDIRECT
                        ;AND NOT DIRECTLY. IF A ERR OCCURES PRESSING CONTINUE
                        ;WILL CAUSE THE PROGRAN TO LLOP ON ERROR UNTILL
                        ;RESTART.

02206 102520  DEF14:  SUBZL 0,0          ;C(0)=1,LOOP ON ERROR SWITCH
02207 030114          LDA 2,BUFF         ;C(2) POINTS TO THE BUFFER
02210 034122          LDA 3,C377         ;C(3)=ITERATION COUNTER
02211 050107          STA 2,TEM          ;STORE BUFF POINTER IN C(TEM)
02212 150000          COM 2,2            ;CHANGE C(2)
02213 060377          NIOP CPU           ;SYNC AT 7-A74
02214 052107          STA 2,@TEM         ;STORE VIA C(TEM) INDIRECT
02215 150000          COM 2,2            ;RESTORE C(2)
02216 024107          LDA 1,TEM          ;DID C(TEM) CHANGE?
02217 132414          SUB# 1,2,SZR       ;C(1)=VALUE IN C(TEM)
02220 063077          HALT               ;C(2)=ORIG VALUE OF TEM
02221 146414          SUB# 2,1,SZR       ;IF C(1) IS COMP OF C(2)
02222 102400          SUB 0,0            ;STA STORED DIRECT
02223 113000          ADD 0,2            ;LOOP ON ERROR
02224 116404          SUB 0,3,SZR        ;ITERATION COUNTER
02225 000764          JMP DEF14+3
02226 101010  DEF15:  MOV# 0,0           ;CHANGE TO LOOP PROGRAM

                        ;A TEST OF INDIRECT ADDRESSING
                        ;THIS ROUTINE WILL TEST THE ABILITY OF THE STA
                        ;INSTRUCTION
                        ;TO STORE IN A BUFFER USING INDIRECT ADDRESSING. THE
                        ;DATA
                        ;STORED IS THE COMPLEMENT OF THE ADDRESS INTO WHICH IT
                        ;IS
                        ;STORED. EXAMPLE: C(5073)=172704. IF A ERROR OCCURES
                        ;PRESS
                        ;CONTINUE TO LOOP ON THE FAILING CONDITIONS.

02227 102520  DEF20:  SUBZL 0,0          ;C(0)=1,LOOP ON ERR SWITCH
02230 030114          LDA 2,BUFF         ;C(2)=FIRST BUFFER LOCATION
02231 050107          STA 2,TEM          ;SET TEM TO POINT TO BUFFER
02232 154000          COM 2,3
02233 060377          NIOP CPU           ;SYNC AT 7-A74
02234 056107          STA 3,@TEM         ;STORE COMP OF ADDRESS
02235 025000          LDA 1,0,2          ;VALUE STORED TO AC1
02236 136414          SUB# 1,3,SZR       ;DIS IT STORE OK?
02237 063077          HALT               ;C(2)=ADDRESS IN BUFFER
02240 136414          SUB# 1,3,SZR       ;C(3)=DATA STORED
02241 102400          SUB 0,0            ;C(1)=DATA FROM BUFFER
02242 113000          ADD 0,2            ;ADVANCE TO NEXT LOC
02243 034115          LDA 3,FIN
02244 156414          SUB# 2,3,SZR       ;TEST FOR END OF BUFFER
02245 000764          JMP DEF20+2
02246 101010  DEF21:  MOV# 0,0           ;CHANGE TO LOOP TEST
```

```
^^^

                ;A TEST OF INDIRECT ADDRESSING
                ;THE BUFFER IS SET TO CONTAIN A ADDRESS PATTERN. A LDA
                ;INSTRUCTION VIA C(2) THEN CAUSES A INDIRECT REFFERANCE
                ;TO BE MADE AT EACH LOCATION IN THE BUFFER. BECAUSE
                ;EACH LOCATION CONTAINS ITS ADDRESS THE LDA INSTRUCTION
                ;WILL LOAD THE LOCATION INDIRECTED THROUGH. PRESSING
                ;CONTINUE AFTER A ERR WILL CAUSE THE PROGRAM TO LOOP ON
                ;ERROR UNTILL RESTARTED.

02247 102520    DEF30:  SUBZL 0,0       ;C(0)=1,LOOP ON ERROR SWITCH
02250 030114            LDA 2,BUFF      ;FIRST ADDRESS IN BUFFER
02251 034115            LDA 3,FIN       ;FINAL ADDRESS IN BUFFER
02252 051000            STA 2,0,2       ;STORE A ADDRESS PATTERN
02253 151400            INC 2,2         ;IN THE BUFFER
02254 156414            SUB# 2,3,SZR    ;EXAMPLE: C(7314)=7314
02255 000775            JMP .-3

02256 030114    DEF31:  LDA 2,BUFF      ;FIRST BUFFER LOCATION
02257 060377            NIOP CPU        ;SYNC AT 7-A74
02260 027000            LDA 1,@0,2      ;SHOULD LOAD WD POINTED TO
02261 132414            SUB# 1,2,SZR    ;C(2)=ADDRESS INDIRECT THROUGH
02262 063077            HALT            ;C(1)=VALUE OBTAINED
02263 132414            SUB# 1,2,SZR
02264 102400            SUB 0,0         ;CLEAR 0 TO LOOP ON ERROR
02265 113000            ADD 0,2         ;INC TO NEXT BUFFER LOC
02266 156414            SUB# 2,3,SZR    ;TEST FOR END OF BUFFER
02267 000770            JMP DEF31+1
02270 101010    DEF32:  MOV# 0,0        ;CHANGE TO LOOP TEST

                ;A TEST OF INDIRECT ADDRESSING
                ;A MODIFIED ADDRESS PATTERN (C(ADDRESS)=ADDRESS-1) IS
                ;STORED IN THE BUFFER. A STA INSTRUCTION INDIRECTS
                ;THROUGH EACH LOCATION IN THE BUFFER STORING IN THAT
                ;LOCATION-1. PRESSING CONTINUE AFTER A ERROR CAUSES THE
                ;PROGRAM TO LOOP ON THE ERROR UNTILL RESTARTED.

02271 102520    DEF34:  SUBZL 0,0       ;C(0)=1
02272 034115            LDA 3,FIN       ;FINAL LOCATION IN BUFFER
02273 030114            LDA 2,BUFF      ;FIRST LOCATION IN BUFFER
02274 051001            STA 2,1,2       ;IN EACH BUFFER LOCATION
02275 151400            INC 2,2         ;STORE THE ADDRESS OF THAT
02276 156414            SUB# 2,3,SZR    ;LOCATION-1
02277 000775            JMP .-3
02300 030114    DEF35:  LDA 2,BUFF      ;FIRST BUFFER LOCATION
02301 060377            NIOP CPU        ;SYNC AT 7-A74
02302 053001            STA 2,@1,2      ;DEFER THROUGH BUFFER,STORE
02303 025000            LDA 1,0,2       ;C(2) IN BUFFER-1.THEREFOR
02304 132414            SUB# 1,2,SZR    ;STORE ADDRESS IN ADDRESS.
02305 063077            HALT            ;FOR EXAMPLE: C(4050)=XXX
02306 132414            SUB# 1,2,SZR    ;C(4051)=4050,C(2)=4050
02307 102400            SUB 0,0         ;INDIRECT THROUGH 4051 TO
02310 113000            ADD 0,2         ;STORE C(2) IN LOCATION 4050
02311 156014            ADC# 2,3,SZR    ;TEST FOR END OF BUFFER
02312 000767            JMP DEF35+1
02313 101010    DEF36:  MOV# 0,0        ;CHANGE TO LOOP TEST
```

^^^

```
                ;A TEST OF INDIRECT ADDRESSING (2 LEVELS)
                ;THE BUFFER IS SET TO CONTAIN A ADDRESS PATTERN. A LDA
                ;INSTRUCTION USING TWO LEVELS OF INDIRECT ADDRESSING
                ;THEN REFFERANCES EACH LOCATION IN THE BUFFER. PRESSING
                ;CONTINUE AFTER A ERROR WILL CAUSE THE PROGRAM TO LOOP
                ;ON THE ERROR UNTILL RESTARTED.

02314 102520  DEF40:  SUBZL 0,0        ;C(0)=1,LOOP ON ERR SWITCH
02315 030114          LDA 2,BUFF       ;FIRST BUFFER LOCATION
02316 034115          LDA 3,FIN        ;FINAL BUFFER LOCATION
02317 051000          STA 2,0,2        ;STORE A ADDRESS PATTERN
02320 151400          INC 2,2          ;IN THE BUFFER
02321 156414          SUB# 2,3,SZR
02322 000775          JMP .-3

02323 030114  DEF41:  LDA 2,BUFF       ;FIRST BUFFER LOCATION
02324 126620          SUBZR 1,1        ;C(1)=100000,THE DEFER BIT
02325 147000          ADD 2,1          ;C(2)=INDIRECT POINTER
02326 044107          STA 1,TEM        ;TO BUFFER
02327 060377          NIOP CPU         ;SYNC AT 7-A74
02330 026107          LDA 1,@TEM       ;GO INDIRECT FOR DATA
02331 132414          SUB# 1,2,SZR     ;C(1)=DATA FROM BUFFER
02332 063077          HALT             ;C(2)=CORRECT DATA
02333 132414          SUB# 1,2,SZR     ;2 LEVELS INDIRECT FAIL
02334 102400          SUB 0,0          ;CLEAR TO LOOP ON ERROR
02335 113000          ADD 0,2          ;       INC TO NEXT BUFF LOC
02336 156414          SUB# 2,3,SZR     ;TEST FOR END OF BUFFER
02337 000765          JMP DEF41+1
02340 101010  DEF42:  MOV# 0,0         ;CHANGE TO LOOP TEST



                ;A TEST OF JSR (HARD TEST)
                ;THE BUFFER IS FILLED WITH THE FOLLOWING
                ;       JSR 2,3          ;GO TO THE LOCATION SPECIFIED BYC(3)+2
                ;       JSR 0,2          ;GO BACK TO THE CONTROL PROGRAM
                ;       JSR 0,2          ;"   "    "   "      "     "
                ;       JSR 0,2          ;"    "   "   "      "     "
                ;       JSR 2,3          ;GO TO HHE LOCATION SPECIFIED BY C(3)+2
                ;THE C(3) IS INITIALLY SET TO BUFFER+1
                ;THE C(2) "  "          "   "  RETURN TO MAIN PROGRAM
                ;THE PROGRAM THEN JUMPS TO THE FIRST BUFFER LOCATION.
                ;THE BUFFER SEQUENCE IS AS FOLLOWS:
                ;#1     JSR TO BUFFER+3 C(3)=BUFFER+1
                ;#2     JSR TO BUFFER+3 C(3)=BUFFER+4
                ;#3     JSR TO BUFFER+6 C(3)=BUFFER+4
                ;#4     JSR TO BUFFER+6 C(3)=BUFFER+7
                ;#5     JSR TO BUFFER+11 C(3)=BUFFER+7
                ;#6     ETC
```

```
AAA

                ;THIS PROCESS CONTIMUES UNTILL THE END OF THE BUFFER IS
                ;REACHED.
                ;AT FINAL ADDEESS,FINAL-1,FINAL+1,THE PROGRAM RETURNS
                ;TO THE
                ;MAIN SEQUENCE. THE PC STORED BY THE FINAL JSR IS
                ;CHECKED.

02341 030114  JSR20: LDA 2,BUFF       ;FIRST BUFFER LOCATION
02342 034115         LDA 3,FIN        ;LAST  BUFFER LOCATION
02343 020422         LDA 0,CJSR3      ;CONSTANT (JSR 2,3)=CONTINUE
02344 024420         LDA 1,CJSR2      ;CONSTANT (JSR 0,2)=RETURN
02345 041000  JSR21: STA 0,0,2        ;STORE CONTINUE
02346 045001         STA 1,1,2        ;"     RETURN
02347 045002         STA 1,2,2        ;"     RETURN
02350 151400         INC 2,2
02351 151400         INC 2,2
02352 151400         INC 2,2
02353 156432         SUBZ# 2,3,SZC    ;TEST FOR END OF BUFFER
02354 000771         JMP JSR21        ;PERHAPS 1,2 EXTRA WORDS
02355 045375         STA 1,-3,2       ;MAKE LAST CONTINUE A RETURN

02356 034114  JSR22: LDA 3,BUFF       ;INITIAL SETUP OF C(3)
02357 175400         INC 3,3          ;FOR FIRST JSR IN BUFFER
02360 030406         LDA 2,CJSR4      ;THE PROGRAM RETURN
02361 102400         SUB 0,0          ;BOTH C(0),C(1) SET TO 0
02362 126400         SUB 1,1          ;WILL CHECK FOR NO EFFECT BY
                                      ;JSR
02363 001777         JMP -1,3         ;GO TO THE BUFFER!

02364 005000  CJSR2: JSR 0,2          ;THREE CONSTANTS
02365 005402  CJSR3: JSR 2,3
02366 002367  CJSR4: .+1
02367 107004  JSR23: ADD 0,1,SZR     ;C(0),C(1) SHOULD NOT HAVE
02370 063077         HALT             ;BEEN CHANGED!!
02371 020115         LDA 0,FIN        ;TEST C(3) FOR FINAL,FINAL-1
02372 162644         SUBOR 3,0,SZR    ;OR FINAL+1,ADDRESS
02373 100005         COM 0,0,SNR      ;C(3) = ADDRESS OF RETURNING
02374 101011         MOV# 0,0,SKP
02375 063077         HALT
02376 101000         MOV 0,0          ;CHANGE TO LOOP TEST
```

^^^

```
                ;A TEST OF INDIRECT ADDRESSING (2 LEVELS)
                ;THE FIRST HALF OF THE BUFFER IS FILLED WITH (JSR 0,3)
                ;INSTRUCTIONS. THE SECOND HALF IS FILLED
                ;AND ADDRESS POINTING TO THE (JSR) INSTRUCTIONS IN THE
                ;BOTTEM. C(TEM) POINTS TO THE TOP HALF OF THE BUFFER.
                ;THE SEQUENCE IS AS FOLLOWS:
                ;#1      GO INDIRECT THROUGH REGISTER TEM
                ;#2      WORD IN TOP POINTS TO BOTTEM
                ;#3      INSTRUCTION IN BOTTEM IS EXECUTED
                ;#4      PC (C(3)) FROM BOTTEM IS CHECKED
                ;PRESSING CONTINUE AFTER A ERROR WILL CAUSE THE PROGRAM
                ;TO LOOP ON THE ERROR UNTILL RESTARTED.

02377 034115  DEF50:  LDA 3,FIN       ;FINAL BUFFER ADDRESS
02400 030114          LDA 2,BUFF      ;FIRST BUFFER ADDRESS
02401 024114          LDA 1,BUFF
02402 020437          LDA 0,CJDF1     ;CONSTANT (JSR 0,3)
02403 166640          SUBOR 3,1       ;C(1)=(LAST-FIRST)/2=-WC
02404 041000          STA 0,0,2       ;STORE THE JSRS'
02405 151400          INC 2,2         ;IN BOTTEM PART OF BUFFER
02406 125404          INC 1,1,SZR
02407 000775          JMP .-3
02410 050107          STA 2,TEM       ;C(TEM)=FIRST LOC IN TOP
02411 020114          LDA 0,BUFF
02412 041000  DEF51:  STA 0,0,2       ;STORE A MODIFIED ADDRESS
02413 101400          INC 0,0         ;PATTERN IN TOP HALF OF
02414 151400          INC 2,2         ;THE BUFFER. C(TOP)=
02415 156414          SUB# 2,3,SZR    ;BOTTEM ADDRESS+100000
02416 000774          JMP .-4

02417 030107  DEF52:  LDA 2,TEM       ;C(2) POINTS TO TOP HALF
02420 102620          SUBZR 0,0       ;C(0)=100000
02421 143000          ADD 2,0         ;THIS ROUINE AADDS THE
02422 040107          STA 0,TEM       ;INDIRECT BIT TO C(TEM)
02423 024114          LDA 1,BUFF      ;C(1) USED FOR CHECK
02424 102520          SUBZL 0,0       ;C(0)=1,LOOP ON ERR SWITCH

02425 060377  DEF53:  NIOP CPU        ;SYNC AT 7-A74
02426 006107          JSR @TEM        ;GO TO THE BUFFER C(3)=
02427 136014          ADC# 1,3,SZR    ;JSR RETURN.C(1)=CORRECT
02430 063077          HALT            ;FINAL ADDRESS.INDIRECT WITH
02431 136014          ADC# 1,3,SZR    ;JSR FAILED
02432 102400          SUB 0,0         ;CLEAR TO LOOP ON ERROR
02433 107000          ADD 0,1         ;POINT TO NEXT BUFFER LOC
02434 101014          MOV# 0,0,SZR
02435 010107          ISZ TEM         ;INC IF NOT ERROR
02436 132014          ADC# 1,2,SZR    ;TEST FOR END OF BUFFER
02437 000766          JMP DEF53
02440 101011  DEF54:  MOV# 0,0,SKP    ;CHANGE TO LOOP TEST
02441 005400  CJDF1:  JSR 0,3         ;A CONSTANT
```

^^^

```
02442 020114  RANFL: LDA 0,BUFF       ;FILL THE BUFFER WITH
02443 040107         STA 0,TEM        ;RANDOM ADDRESS IN
02444 034115         LDA 3,FIN        ;BITS 4-15. THIS VALUE
02445 024000         LDA 1,0          ;WILL POINT TO SOME
02446 121000  RANFC: MOV 1,0          ;OTHER SPOT IN THE BUFFER
02447 024433         LDA 1,C1347      ;BIT 0 IS RANDOMLY 0/1
02450 152620         SUBZR 2,2
02451 107222         ADDZR 0,1,SZC    ;THIS
02452 147000         ADD 2,1
02453 131000         MOV 1,2          ;ROUTINE PRODUCES A
02454 113520         ANDZL 0,2        ;RANDOM 16 BIT NUMBER
02455 107000         ADD 0,1          ;C(1)=RANDOM
02456 146400         SUB 2,1

02457 020112         LDA 0,MSIZE      ;MASK OF MEM SIZE
02460 123400         AND 1,0          ;C(0)=RANDOM MODULO MEMORY
02461 030114         LDA 2,BUFF
02462 162433         SUBZ# 3,0,SNC    ;IS C(0) WITH IN THE
02463 142033         ADCZ# 2,0,SNC    ;RANGE OF THE BUFFER?
02464 000762         JMP RANFC        ;NO TO BIG OR TO SMALL
02465 030107         LDA 2,TEM
02466 112415         SUB# 0,2,SNR
02467 000757         JMP RANFC        ;REJECT IF ADDRESS SAME
02470 101120         MOVZL 0,0
02471 125100         MOVL 1,1         ;C(BIT 0) RANDOM DETERMINES
02472 101200         MOVR 0,0         ;IF WORD GETS INDIRECT BIT
02473 125200         MOVR 1,1
02474 042107         STA 0,@TEM       ;STORE THE WORD
02475 010107         ISZ TEM
02476 030107         LDA 2,TEM        ;TEST FOR END OF BUFFER
02477 156014         ADC# 2,3,SZR
02500 000746         JMP RANFC        ;MAKE MORE RANDOM
02501 101011         MOV# 0,0,SKP     ;CHANGE TO LOOP
02502 135753  C1347: 135753
```

^^^

```
                ;A TEST OF INDIRECT ADDRESSING
                ;THE BUFFER IS FILLED WITH PSEUDO RANDOM NUMBERS WHICH
                ;ARE ADDRESSES. THE VALUE OF EACH ADDRESS POINTS TO
                ;SOME SPOT IN THE BUFFER. BIT 0 OF EACH BUFFER
                ;WORD IS RANDOMLY SET OR CLEARED. THE PROGRAM FIRST
                ;POINTS TO THE FIRST BUFFER LOCATION. THE FINAL
                ;EFFECTIVE ADDRESS IS CALCULATED UP TO 15 LEVELS OF
                ;INDIRECT. A LDA INSTRUCTION THEN REFFERANCES
                ;INDIRECTLY
                ;THE BUFFER. THE RESULT OC THE INDIRECT LDA SHOULD BE
                ;THE SAME AS THE CALCULATED VALUE. IF IT IS NOT THE
                ;SAME THE CORRECT CHAIN SHOULD BE CALCULATED WITH PAPER
                ;AND PENCIL.RESTARTING THE PROGRAM AT THE LOCATION
                ;FOLLOWING THE HALT WILL CAUSE THE PROGRAM TO ENTER
                ;A FAILING LOOP UNTILL RESTARTED. A SYNC PULSE AT
                ;7-A74 PROCEEDS THE INDIRECT CHAIN.

02503 024114  DEF60:  LDA 1,BUFF        ;FIRST LOCATION IN THE BUFFER
02504 121120          MOVZL 1,0
02505 101240          MOVOR 0,0
02506 040123          STA 0,CF00
02507 102520          SUBZL 0,0         ;C(0)=1,LEVEL COUNTER
02510 131000          MOV 1,2
02511 101125          MOVZL 0,0,SNR     ;COUNT INDIRECT LEVELS
02512 000417          JMP DEF62         ;15 LEVELS,GIVE UP THIS CHAIN
02513 031000          LDA 2,0,2         ;GET BUFFER WORD
02514 151132          MOVZL# 2,2,SZC    ;DOES IT HAVE A INDIRECT BIT
02515 000774          JMP .-4           ;YES KEEP LOOKING
02516 050107          STA 2,TEM
02517 031000          LDA 2,0,2         ;C(2)=FINAL VALUE

02520 102520  DEF61:  SUBZL 0,0         ;C(0)=1,LOOP ON ERR SWITCH
02521 060377          NIOP CPU          ;SYNC AT 7-A74
02522 036123          LDA 3,@CF00       ;REFFERANCE THE BUFFER
02523 172414          SUB# 3,2,SZR      ;C(3)=WORD OBTAINED
02524 063077          HALT              ;C(2)=CORRECT
02525 172414          SUB# 3,2,SZR      ;C(1)=PLACE BUFFER WAS ENTERED
02526 102401          SUB 0,0,SKP       ;WRITE DOWN THE CORRECT
02527 101005          MOV 0,0,SNR       ;INDIRECT CHAIN,SINGLE STEP
02530 000771          JMP DEF61+1       ;LOOP ON THE ERROR

02531 125400  DEF62:  INC 1,1           ;GO TO THE NEXT BUFFER LOC
02532 034115          LDA 3,FIN         ;TEST FOR FINAL
02533 136414          SUB# 1,3,SZR
02534 000750          JMP DEF60+1       ;NOT YET BUFFER END

02535 101010  DEF63:  MOV# 0,0          ;CHANGE TO LOOP TEST
```

^^^

```
                ;A TEST OF INDIRECT ADDRESSING
                ;THIS TEST IS MIMULAR TO THE PREVIOUS TEST IN THAT THE
                ;BUFFER IS FILLED WITH PSEUDO RANDOM NUMBERS. EACH
                ;LOCATION IN THE BUFFER MAY CONTAIN A INDIRECT BIT.
                ;EACH LOCATION ALSO CONTAINS A N BIT ADDRESS OF A
                ;BUFFER LOCATION. A ROUTINE CALCULATES EFFECTIVE
                ;ADDRESS UP TO 15 LEVELS. THE EFFECT OF A INDIRECT
                ;ISZ/DSZ INSTRUCTION PAIR IS CHECKED.

02536 030114  EXCH:   LDA 2,BUFF        ;CAN'T USE AUTO INC/DEC YET
02537 034115          LDA 3,FIN         ;EXCHANGE BUFFER ENDS SO
02540 021000          LDA 0,0,2         ;IT WONT BE THE SAME AS
02541 025777          LDA 1,-1,3        ;LAST TEST
02542 045000          STA 1,0,2
02543 041777          STA 0,-1,3
02544 151400          INC 2,2
02545 174400          NEG 3,3           ;INC TOWARDS TOP,DEC TOWARDS
02546 174000          COM 3,3           ;BUTTEM
02547 172433          SUBZ# 3,2,SNC     ;TEST FOR MIDDLE
02550 000770          JMP EXCH+2        ;OF THE BUFFER

02551 024114  DEF64:  LDA 1,BUFF        ;FIRST BUFFER LOCATION
02552 121120          MOVZL 1,0
02553 101240          MOVOR 0,0
02554 040123          STA 0,CF00
02555 102520          SUBZL 0,0         ;C(0)=LEVEL COUNTER
02556 131000          MOV 1,2
02557 101125          MOVZL 0,0,SNR     ;IF MORE THAN 15 LEVELS
02560 000422          JMP DEF66+2       ;OF INDIRECT,GIVE UP
02561 031000          LDA 2,0,2         ;CALCULATE EFFECTIVE
02562 151132          MOVZL# 2,2,SZC    ;ADDRESS. LOOK FOR INDIRECT
02563 000774          JMP .-4           ;BIT 0 A ONE
02564 050107          STA 2,TEM         ;C(TEM)=EFFECTIVE ADDRESS
02565 031000          LDA 2,0,2         ;C(2)=DATA THAT ADDRESS
```

^^^

```
02566 102520  DEF65:  SUBZL 0,0          ;C(0)=1,LOOP ON ERROR SWITCH
02567 060377          NIOP CPU           ;SYNC AT 7-A74
02570 012123          ISZ @CFOO
02571 036107          LDA 3,@TEM
02572 016107          DSZ @TEM
02573 156015          ADC# 2,3,SNR       ;INCREMENT CORRECT?
02574 000413          JMP DEF67          ;C(2)=CORRECT,C(TEM)=ADDRESS
02575 063077          HALT               ;C(3)=WORD FROM BUFFER
02576 102400          SUB 0,0            ;CLEAR TO LOOP ON ERROR
02577 052107          STA 2,@TEM         ;HOPEFULLY RESTORE MEMORY
02600 101005  DEF66:  MOV 0,0,SNR        ;TEST THE ERR SWITCH
02601 000766          JMP DEF65+1        ;LOOP ON ERROR
02602 125400          INC 1,1            ;ADVANCE BUFFER POINTER
02603 034115          LDA 3,FIN          ;TEST FOR FINAL ADDRESS
02604 136414          SUB# 1,3,SZR
02605 000745          JMP DEF64+1
02606 000406          JMP .+6            ;GO TO NEXT TEST

02607 036107  DEF67:  LDA 3,@TEM         ;ISZ WORKED TEST DSZ
02610 172415          SUB# 3,2,SNR       ;C(2)=CORRECT,C(TEM)=ADDRESS
02611 000767          JMP DEF66          ;C(3)=DATA FROM MEMORY
02612 063077          HALT               ;PRESS CONTINUE TO
02613 000763          JMP DEF66-2        ;RESTORE AND LOOP ON ERR
02614 101010          MOV# 0,0           ;CHANGE TO LOOP TEST


                      .EOT
```

```
                ;A TEST OF AUTO INCREMENT/DECREMENT
                ;THIS TEST WILL INSURE THAT REGISTERS 20-37
                ;DO NOT CHANGE VALUE WHEN REFFERANCED WITHOUT A
                ;INDIRECT BIT.

02615 102400    ID0:    SUB 0,0             ;FIRST PASS STORES (0)
02616 152400            SUB 2,2             ;IN REGISTERS 20-37
02617 126620            SUBZR 1,1           ;SECOND PASS STORES (-1)
02620 041020            STA 0,20,2
02621 151400            INC 2,2             ;INC ADDRESS
02622 125224            MOVZR 1,1,SZR       ;ITERATION COUNT OF 20
02623 000775            JMP .-3

02624 176620    ID1:    SUBZR 3,3           ;SET ITERATION COUNT
02625 021000            LDA 0,0,2           ;LOOK FOR A CHANGE
02626 025000            LDA 1,0,2           ;IN TO REFFERANCES
02627 106414            SUB# 0,1,SZR        ;TO AUTO INC/DEC LOCATIONS
02630 063077            HALT                ;C(2)=ADDRESS,C(0)=FIRST
02631 151400            INC 2,2             ;REFFERANCE,C(1)=SECOND
02632 175224            MOVZR 3,3,SZR       ;GO THROUGH ALL INC/DEC
02633 000772            JMP ID1+1           ;REGISTERS

02634 102000            ADC 0,0             ;C(0)=-1
02635 151112            MOVL# 2,2,SZC       ;FIRST OR SECOND PASS
02636 000403            JMP ID2             ;SECOND,EXIT TEST
02637 152620            SUBZR 2,2           ;C(2)=100000
02640 000757            JMP ID0+2
02641 101010    ID2:    MOV# 0,0            ;CHANGE TO LOOP TEST
```

^^^

```
02642 020124   ID4:    LDA 0,C174X       ;CHECK THAT LOCATIONS
02643 024112           LDA 1,MSIZE       ;4027,2027,1027 ETC ARE
02644 176620           SUBZR 3,3         ;NOT AUTO INCREMENT LOC-
02645 175220           MOVZR 3,3         ;ATIONS.
02646 137415           AND# 1,3,SNR
02647 000776           JMP .-2

02650 055427           STA 3,27,3        ;STORE SOME DATA
02651 027427           LDA 1,@27,3
02652 025427           LDA 1,27,3        ;DID IT CHANGE VIA
02653 136414           SUB# 1,3,SZR      ;THE INDIRECT REFFERANCE?
02654 000410           JMP ID5           ;YES IT FAILED.
02655 024113           LDA 1,C40         ;NO ITS OK.
02656 175220           MOVZR 3,3
02657 137415           AND# 1,3,SNR
02660 000770           JMP ID4+6
02661 101404           INC 0,0,SZR
02662 000761           JMP ID4+1
02663 000407           JMP ID6+1

02664 063077   ID5:    HALT              ;C(3)=DATA STORED,C(1)=
02665 055427           STA 3,27,3        ;DATA AFTER INDIRECT.
02666 060377           NIOP CPU          ;SYNC 7-A74
02667 027427           LDA 1,@27,3       ;BMA INPUT TO AND GATE
02670 025427           LDA 1,27,3        ;FAILED.
02671 000774   ID6:    JMP ID5+1
```

^^^

```
                ;AUTO INCREMENT CHECK
                ;COUNT ALL THE AUTO INCREMENT REGISTERS FROM
                ;60000-77777
                ;CHECK THE VALUE AGAINST A ADD INSTRUCTION. PRESS
                ;CONTINUE AFTER A HALT TO LOOP ON ERROR.

02672 102520  ID10:   SUBZL 0,0         ;C(0)=1,LOOP ON ERROR SWITCH
02673 176400          SUB 3,3           ;C(3)=0-7,REGISTER POINTER
02674 152620          SUBZR 2,2
02675 151240          MOVOR 2,2
02676 151220          MOVZR 2,2         ;STARTING VALUE OF 60000
02677 051420          STA 2,20,3        ;SETUP AUTO INC REGISTER
02700 060377          NIOP CPU          ;SYNC AT 7-A74
02701 027420          LDA 1,@20,3       ;THIS SHOULD INC THE REG
02702 025420          LDA 1,20,3        ;C(1)=VALUE OF AUTO REG
02703 146014          ADC# 2,1,SZR      ;C(2)=CORRECT VALUE-1
02704 063077          HALT              ;C(3)+20=FAILING REG
02705 146014          ADC# 2,1,SZR
02706 102400          SUB 0,0           ;CLEAR TO LOOP ON ERROR
02707 113000          ADD 0,2
02710 151113          MOVL# 2,2,SNC     ;TEST FOR COUNT OF 100000
02711 000766          JMP ID10+5        ;NOT YET
02712 175400          INC 3,3           ;ADVANCE TO NEXT AUTO REG
02713 024076          LDA 1,C10
02714 136414          SUB# 1,3,SZR      ;TEST FOR LAST REGISTER
02715 000757          JMP ID10+2
02716 101010  ID11:   MOV# 0,0          ;CHANGE TO LOOP TEST

                ;AUTO DECREMENT CHECK
                ;COUNT ALL THE AUTO DECREMENT REGISTERS FROM 20000-0
                ;CHECK TEH VALUE AGAINST A ADD INSTRUCTION. PRESS
                ;CONTINUE AFTER A ERROR TO SCOPE THE FAILURE

02717 102520  ID14:   SUBZL 0,0         ;C(0)=1,LOOP ON ERROR SWIT
02720 176400          SUB 3,3           ;C(3)=0-7,REGISTER POINTER
02721 152620          SUBZR 2,2
02722 151220          MOVZR 2,2
02723 151220          MOVZR 2,2         ;STARTING VALUE OF 20000
02724 051430          STA 2,30,3        ;SETUP AUTO DECREMENT REG
02725 060377          NIOP CPU          ;SYNC AT 7-74
02726 027430          LDA 1,@30,3       ;SHOULD DECREMENT THE REG
02727 025430          LDA 1,30,3        ;C(1)=VALUE OF AUTO DEC REG
02730 132014          ADC# 1,2,SZR      ;C(2)=CORRECT VALUE+1
02731 063077          HALT              ;C(3)+30=FAILING AUTO REG
02732 132014          ADC# 1,2,SZR
02733 102400          SUB 0,0           ;CLEAR TO LOOP TEST
02734 112404          SUB 0,2,SZR       ;TEST FOR ZERO COUNT
02735 000767          JMP ID14+5
02736 175400          INC 3,3           ;ADVANCE TO NEXT AUTO REG
02737 024076          LDA 1,C10         ;TEST FOR LAST AUTO
02740 136414          SUB# 1,3,SZR      ;DECREMENT REGISTER
02741 000760          JMP ID14+2
02742 101010  ID15:   MOV# 0,0          ;CHANGE TO LOOP TEST
```

^^^

```
                ;A TEST OF AUTO INCREMENT
                ;EACH LOCATION IN THE BUFFER IS SET TO A ADDRESS
                ;PATTERN=1
                ;VIA THE AUTO INDEX REGISTERS. THE REGISTERS COUNTING
                ;ABILITY AS WELL AS THE DATA IS CHECKED.THE PROGRAM
                ;WILL LOOP ON ERROR IF CONTINUED AFTER A HALT.

02743 102000            ADC 0,0
02744 176400  ID20:     SUB 3,3
02745 030114            LDA 2,BUFF      ;FIRST BUFFER LOCATION
02746 051420            STA 2,20,3      ;SETUP AUTO INC REGISTER
02747 060377            NIOP CPU        ;SYNC AT 7-A74
02750 053420            STA 2,@20,3     ;STORE C(2) VIA AUTO INC
02751 025420            LDA 1,20,3      ;C(AUTO) TO C(1)
02752 146015  ID21:     ADC# 2,1,SNR    ;CHECK INCREMENT FEATURE
02753 000415            JMP ID23        ;ITS OK
02754 063077            HALT            ;C(1)=VALUE OF AUTO INC REG
02755 102400            SUB 0,0         ;C(2)=CORRECT VALUE-1
02756 041000  ID22:     STA 0,0,2       ;C(3)+20=AUTO REGISTER INVOLVED
02757 112400            SUB 0,2         ;C(BUFFER) DESTROYED AFTER TEST
02760 024115            LDA 1,FIN       ;C(1)=FINAL ADDRESS
02761 132414            SUB# 1,2,SZR    ;TEST FOR END OF BUFFER
02762 000764            JMP ID20+2
02763 175400            INC 3,3         ;GO TO NEXT AUTO REGISTER
02764 024076            LDA 1,C10       ;TEST FOR LAST ONE
02765 136414            SUB# 1,3,SZR
02766 000757            JMP ID20+1
02767 000406            JMP ID24        ;GO TO NEXT TEST

02770 025001  ID23:     LDA 1,1,2       ;AUTO INC VALUE OK BUT
02771 132415            SUB# 1,2,SNR    ;DATA STORED IS WRONG
02772 000764            JMP ID22        ;C(1)=DATA FROM MEMORY
02773 063077            HALT            ;C(2)=CORRECT VALUE
02774 000761            JMP ID22-1      ;C(3)+20=AUTO REGISTER
02775 101010  ID24:     MOV# 0,0        ;CHANGE TO LOOP TEST
```

^^^

```
                ;A TEST OF AUTO DECREMENT
                ;EACH LOCATION IN THE BUFFER IS SET TO A ADDRESS
                ;PATTERN+1 VIA THE AUTO DECREMENT REGISTERS. THE
                ;REGISTERS COUNTING ABILITY AS WELL AS THE DATA IS
                ;CHECKED.THE PROGRAM WILL LOOP ON ERROR IF CONTINUED
                ;AFTER A HALT

02776 102000  ID30:   ADC 0,0          ;C(0)=-1,LOOP ON ERR SWITCH
02777 176400          SUB 3,3          ;C(3)=0,FIRST INDEX USED
03000 030115          LDA 2,FIN        ;FINAL BUFFER ADDRESS
03001 051430          STA 2,30,3       ;TO AUTO DECREMENT REGISTER
03002 060377          NIOP CPU         ;SYNC AT 7-A74
03003 053430          STA 2,@30,3      ;STORE C(2) IN BUFFER VIA
03004 025430          LDA 1,30,3       ;C(1)=VALUE OF AUTO DEC REG
03005 132015  ID31:   ADC# 1,2,SNR     ;C(2)=CORRECT VALUE+1
03006 000415          JMP ID33         ;C(3)+30=AUTO REGISTER
03007 063077          HALT             ;AUTO DECREMENT FAILED
03010 102400          SUB 0,0          ;CLEAR TO LOOP ON ERROR
03011 041000  ID32:   STA 0,0,2        ;CLEAR OUT THE PATTERN
03012 113000          ADD 0,2          ;C(2)=C(2)-1
03013 024114          LDA 1,BUFF       ;TEST FOR BEGIN OF BUFFER
03014 132414          SUB# 1,2,SZR
03015 000764          JMP ID30+3
03016 175400          INC 3,3          ;ADVANCE TO NEXT AUTO REG
03017 024076          LDA 1,C10
03020 136414          SUB# 1,3,SZR     ;TEST FOR END
03021 000757          JMP ID30+2       ;AGAIN WITH THIS REGISTER
03022 000406          JMP ID34         ;GO TO NEXT TEST

03023 025377  ID33:   LDA 1,-1,2       ;AUTO DEC VALUE IS OK
03024 132415          SUB# 1,2,SNR     ;BUT THE DATA STORED IS NOT
03025 000764          JMP ID32         ;C(1)=DATA FROM BUFFER
03026 063077          HALT             ;C(2)=CORRECT VALUE
03027 000761          JMP ID32-1       ;C(3)+30=AUTO REGISTER

03030 101010  ID34:   MOV# 0,0         ;CHANGE TO LOOP TEST
```

^^^

```
                        ;A TEST OF AUTO INC USING ISZ
                        ;THE PROGRAM STORES A (-1) IN THE BUFFER.
                        ;A ISZ INSTRUCTION VIA AUTO INCREMENT REGISTER 27
                        ;THEN INCREMENTS THE BUFFER. IF A ERROR SHOULD OCCURE
                        ;PRESSING CONTINUE WILL PLACE THE PROGRAM IN A FAILING
                        ;LOOP UNTILL RESTARTED.

03031 102520  ID40:   SUBZL 0,0          ;C(0)=1,LOOP ON ERROR SWITCH
03032 126000          ADC 1,1            ;C(1)=-1
03033 030114          LDA 2,BUFF         ;FIRST LOCATION IN THE BUFFER
03034 034115          LDA 3,FIN          ;FINAL LOCATION IN THE BUFFER
03035 050027  ID41:   STA 2,27           ;SETUP AUTO INCREMENT REG
03036 045001          STA 1,1,2          ;STORE (-1) IN MEMORY
03037 060377          NIOP CPU           ;SYNC AT 7-74
03040 012027          ISZ @27            ;INCREMENT AND SKIP
03041 102401          SUB 0,0,SKP        ;ISZ FAIL TO SKIP
03042 113001          ADD 0,2,SKP        ;INC TO NEXT LOC
03043 063077          HALT               ;EXAMINE LOC 27
03044 156414          SUB# 2,3,SZR       ;TEST FOR END OF BUFFER
03045 000770          JMP ID41
03046 101010  ID42:   MOV# 0,0           ;CHANGE TO LOOP TEST

                        ;A TEST OF AUTO INC/DEC
                        ;VIA REGISTER 34 THE BUFFER IS FILLED WITH JSR INST.
                        ;A JSR VIA REGISTER 25 IS EXECUTED,THE RETURN FROM
                        ;THE BUFFER IS CHECKED. PRESS CONTINUE TO LOOP ERROR.

03047 024425  ID44:   LDA 1,CIDJ         ;A CONSTANT (JSR 0,3)
03050 034114          LDA 3,BUFF         ;FIRST BUFFER LOCATION
03051 030115          LDA 2,FIN          ;FINAL BUFFER LOCATION
03052 050034          STA 2,34           ;C(34) POINTS TO THE BUFFER
03053 156400          SUB 2,3            ;C(3)= -WORDS IN BUFFER
03054 046034          STA 1,@34          ;FILL BUFFER WITH (JSR)
03055 175404          INC 3,3,SZR
03056 000776          JMP .-2

03057 102520  ID45:   SUBZL 0,0          ;C(0)=1,LOOP ON ERROR SWITCH
03060 024114          LDA 1,BUFF
03061 044025          STA 1,25           ;SETUP AUTO INDEX REG
03062 006025          JSR @25            ;GO TO THE BUFFER
03063 174400          NEG 3,3            ;EFFECTIVELY SUBTRACT 1
03064 174000          COM 3,3            ;FROM C(3)
03065 136014          ADC# 1,3,SZR       ;C(3)=POINT BUFFER ENTERED
03066 102401          SUB 0,0,SKP        ;C(2)=CORRECT-1
03067 107001          ADD 0,1,SKP        ;C(25)=AUTO INC VALUE
03070 063077          HALT               ;25=AUTO INC REGISTER
03071 132014          ADC# 1,2,SZR       ;TEST FOR END OF BUFFER
03072 000767          JMP ID45+2
03073 101011  ID46:   MOV# 0,0,SKP       ;CHANGE TO LOOP TEST
03074 005400  CIDJ:   JSR 0,3            ;A CONSTANT
```

^^^

```
03075 024114  ID50:   LDA 1,BUFF        ;FIRST LOCATION OF THE BUFF
03076 030115          LDA 2,FIN         ;FINAL LOCATION OF THE BUFF
03077 102520          SUBZL 0,0         ;C(0)=1,LOOP ON ERROR SWIT
03100 050033          STA 2,33          ;SETUP AUTO DECREMENT
03101 006033          JSR @33           ;GO TO THE BUFFER
03102 156414          SUB# 2,3,SZR      ;C(3)=PC FROM BUFFER
03103 102401          SUB 0,0,SKP       ;C(2)=INITAL VALUE OF AUTO
03104 112401          SUB 0,2,SKP       ;DEC REGISTER 33
03105 063077          HALT              ;PROG WILL LOOP ON ERROR
03106 132014          ADC# 1,2,SZR      ;TEST FOR BEGIN OF BUFF
03107 000771          JMP ID50+3


03110 034402  DGCA:   LDA 3,.+2         ;TEST DSZ COUNT IN
03111 152001          ADC 2,2,SKP       ;EACH MEMORY MODULE.
03112 004000          4000              ;C(2)=ERROR SWITCH.
03113 020112          LDA 0,MSIZE
03114 116432          SUBZ# 0,3,SZC
03115 002421          JMP @LAST
03116 102620          SUBZR 0,0
03117 143005  DGCX:   ADD 2,0,SNR
03120 000413          JMP DGCB
03121 041400          STA 0,0,3
03122 060377          NIOP CPU          ;SYNC 7-A74
03123 015400          DSZ 0,3
03124 101010          MOV# 0,0
03125 025400          LDA 1,0,3
03126 122015          ADC# 1,0,SNR      ;C(1)=RESULT OF DSZ
03127 000770          JMP DGCX          ;C(0)=VALUE BEFORE DSZ
03130 152400          SUB 2,2
03131 063077          HALT
03132 000765          JMP DGCX
03133 020757  DGCB:   LDA 0,DGCA+2      ;GO TO NEXT LOCATION
03134 117000          ADD 0,3
03135 000754          JMP DGCA+1
03136 000162  LAST:   BEG1

                      .END
```