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

PROGRAM

Binary Format Punch

TAPES

8K High Core Binary:	091-000020-00
High core binary:	091-000005
Low core binary:	091-000006
ASCII source:	090-000007

ABSTRACT

The Binary Punch Program is a NOVA utility routine that punches operator specified ranges of memory in binary format acceptable as input to the Binary Loader. The program uses either the high-speed paper tape punch or the Teletype ASR punch. It is available in two binary versions which, after loading, reside in different areas of memory. It is also available as an ASCII source tape to enable the user to change the origin and reassemble the program.

093-000001-00

1. REQUIREMENTS

1.1 Memory

1K or larger alterable memory.

1.2 Equipment

Teletype ASR or paper tape punch

1.3 External subroutines

None

1.4 Other

None

2. OPERATING PROCEDURE

2.1 Calling Sequence

The Binary Format Punch program must be loaded in the standard manner (see Binary Loader, 093-000003). To begin execution, enter the proper starting address (as given below) in the data switches, press RESET and then START.

	091-00005	091-000006	Other Versions
Teletype	07400	00400	N
High-Speed Punch	07401	00401	N+1

The program will punch blank tape leader and HALT. The user may then put address arguments in the data switches (see 2.2).

2.2 Input Format

a. Enter the initial address of the block to be punched in the data switches. Press CONTINUE.

b. Enter the final address of the block to be punched in the data switches. Press CONTINUE. The program will now punch the block and HALT.

c. If this was not the last block, proceed from step a. If this was the last block, set data switch Ø and press CONTINUE. The program will punch a few frames of tape and HALT.

d. If the data just punched has a starting address, set it in the data switches (bit 0 = 0) and press CONTINUE, otherwise press CONTINUE (bit 0 = 1). The program will punch the start block, punch trailer, and HALT.

e. Additional tapes may be generated by depressing CONTINUE and proceeding from step a.

2.3 Output Format

A binary tape acceptable as input to the Binary Loader.

2.4 Error Returns

None

2.5 State of Active Registers upon Exit

Not applicable

2.6 Cautions to User

None

3. DISCUSSION

3.1 Algorithms

The routine reads address and control information from the data switches. A PWORD subroutine is used to punch words, two tape frames per word. Checksums are calculated according to the block type.

3.2 Limitations and Accuracy

Not applicable

3.3 Size and Timing

The routine requires 146 (octal) words of storage Its speed is limited by the output device, i.e. the paper tape punch or Teletype.

3.4 References

See the Binary Loader write-up (093-000003) for a description of the binary tape format.

3.5 Flow Diagrams

Not applicable.

4. EXAMPLES AND/OR APPLICATIONS

A standard debugging procedure is to load a binary tape, isolate program bugs, and overwrite (patch) the program locations that are in error. After a number of overwrites have been made, or the routine is debugged, it is convenient to get a new object tape. The Binary Format Punch program enables the user to punch a new binary tape from the current state of memory.

The ASCII source tape (090-000007) is provided to enable the user to create versions of the punch program with different origins.

5. PROGRAM LISTING

A listing of the high core routine follows. The low core routine is identical except that the origin is 00400 instead of 07400.

```
;BINARY FORMAT PUNCHER
;       1. PRESS RESET
;       2. SET SWITCHES=07401(FOR PTP);07400(FOR TTO)
;       3. PRESS START
;       4. PROGRAM WILL PUNCH LEADER AND HALT
;       5. A. IF DONE, SET BITO AND PRESS CONTINUE, GO TO STEP
;6
;          B. IF NOT DONE, SET FIRST ADDRESS AND PRESS CONTINUE
;          C. AT HALT, SET FINAL ADDRESS AND PRESS CONTINUE
;          D. DATA WILL BE PUNCHED , AT HALT GO TO STEP 5
;       6. IF PROGRAM IS TO BE STARTED, SET STARTING
;          ADDRESS AND PRESS CONTINUE, ELSE PRESS CONTINUE.
;          PROGRAM WILL PUNCH TRAILER AND HALT.
;          PRESS CONTINUE TO START OVER AT STEP 4.


     007400          .LOC 7400
07400 000403 START:  JMP .+3          ;GET DEVICE CODE FOR TTO
07401 020541         LDA 0,PTPO       ;GET CODE FOR PTP
07402 000402         JMP .+2
07403 020540         LDA 0,TTOO
07404 062677         IORST
07405 024537         LDA 1,INST
07406 030537         LDA 2,INST+1
07407 107000         ADD 0,1
07410 113000         ADD 0,2
07411 044520         STA 1,PUN
07412 050521         STA 2,PUN+2

07413 004506 LOOP:   JSR BLANK        ;PUNCH BLANKS
07414 177500         -300 <- -50# FOR THE 20K BINARY PUNCH PROGRAM

07415 063077 MAIN:   HALT
07416 070477         READS 2
07417 151112         MOVL* 2,2,SZC    ;TEST BIT 0
07420 000456         JMP DONE         ;ALL DONE
07421 063077         HALT             ;OK, GET FINAL ADDRESS
07422 064477         READS 1
07423 125100         MOVL 1,1
07424 125220         MOVZR 1,1        ;CLEAR BIT 0
07425 050511         STA 2,FIRST
07426 044511         STA 1,LAST
07427 030507 BLOCK:  LDA 2,FIRST
07430 024507         LDA 1,LAST
07431 161000         MOV 2,0
07432 122000         ADC 1,0          ;AC0=COUNT
07433 034502         LDA 3,C20
07434 117033         ADDZ* ^,3,SNC    ;SEE IF COUNT.GE.20(8)
07435 160400         NEG 3,0          ;YES, SET TO 20
07436 155000         MOV 2,3
07437 116400         SUB 0,3
07440 054476         STA 3,FIRST
```

```
07441 040477          STA 0,CNT1
07442 040477          STA 0,CNT2
07443 004450          JSR PWORD         ;PUNCH COUNT
07444 141000          MOV 2,0
07445 004446          JSR PWORD         ;PUNCH ADDRESS


                ;NOW, CALCULATE THE CHECK SUM
07446 020472          LDA 0,CNT1
07447 143000          ADD 2,0
07450 155000          MOV 2,3
07451 025400 CHK:     LDA 1,0,3
07452 123000          ADD 1,0
07453 175400          INC 3,3
07454 010464          ISZ CNT1
07455 000774          JMP CHK
07456 100400          NEG 0,0
07457 004434          JSR PWORD         ;PUNCH THE CHECKSUM
07460 021000          LDA 0,0,2
07461 004432          JSR PWORD         ;PUNCH THE DATA WORD
07462 151400          INC 2,2
07463 010456          ISZ CNT2
07464 000774          JMP .-4           ;CONTINUE
07465 004434          JSR BLANK
07466 177776          -2                ;INTER BLOCK BLANKS
07467 030447          LDA 2,FIRST
07470 024447          LDA 1,LAST
07471 146503          SUBL 2,1,SNC
07472 000735          JMP BLOCK
07473 000722          JMP MAIN

                ;ALL DONE
07474 102520 DONE:    SUBZL 0,0         ;PUNCH A 1
07475 004416          JSR PWORD
07476 063077          HALT              ;GET STARTING ADDRESS
07477 060477          READS 0
07500 101102          MOVL 0,0,SZC
07501 102621          SUBZR 0,0,SKP
07502 101220          MOVZR 0,0
07503 111000          MOV 0,2
07504 004407          JSR PWORD
07505 140000          COM 2,0           ;FORM CHECK-SUM
07506 004405          JSR PWORD         ;AND PUNCH IT
07507 004412          JSR BLANK
07510 177500          -300 ← -50# FOR THE 20K BINARY PUNCH PROGRAM
07511 063077          HALT
07512 000701          JMP LOOP
```

```
                    ;PUNCH A WORD IN AC0
07513 054405 PWORD:  STA 3,SRTN
07514 004415         JSR PUN
07515 101300         MOVS 0,0
07516 004413         JSR PUN
07517 002401         JMP @SRTN
07520 000000 SRTN:   0

07521 054777 BLANK:  STA 3,SRTN
07522 025400         LDA 1,0,3
07523 102400         SUB 0,0
07524 004405         JSR PUN
07525 125404         INC 1,1,SZR
07526 000775         JMP .-3
07527 010771         ISZ SRTN
07530 002770         JMP @SRTN
07531 063513 PUN:    SKPBZ PTP
07532 000777         JMP .-1
07533 061113         DOAS 0,PTP
07534 001400         JMP 0,3
07535 000020 C20:    20
07536 000000 FIRST:  0
07537 000000 LAST:   0
07540 000000 CNT1:   0
07541 000000 CNT2:   0
07542 000013 PTP0:   PTP
07543 000011 TTO0:   TTO
07544 063500 INST:   SKPBZ 0
07545 061100         DOAS 0,0

                     .END
```