Programming Staff Note 20

Computation Center
January 6, 1964

FROM: Janet Allen
SUBJECT: Diagnostic Program for the 7094 CTSS Background

Now available for CTSS background is the I/Ø program TRAPCK, which writes tapes on channels A and B, and then does the same tests as TAPCK (see staff note 14) but with traps enabled, disabled, and inhibited.

This program is in absolute row binary, and is loaded on-line by the two card loader. The time clock must be off (key 29 must be up) for all the diagnostic programs, or the loader will hang up.

TRAPCK, like the other diagnostics, prints on-line, using NPRINT, when sense switch two is down, at the completion of each pass. When any error occurs, the program prints an error indication and halts. Pressing start will cause the program to continue checking. The program runs in an infinite loop, returning to rewrite the tape upon the occurence of some errors.

I. Operating Instructions
TRAPCK begins at location 62 (octal).
1. Deck - 2 card loader
main program
subroutine NPRINT
transfer card
2. Ready the deck in the card reader.
3. Scratch tapes on A4, B1
4. Sense switch two down if printing at end of each pass is desired
5. Keys 21,34, and 35 down to load cards while CTSS operating (key 29 - clock - must be up)
6. Clear and load cards
7. On any halt (HTR *+1) check printer. Press start to continue testing.

II. Program Description

TRAPCK is basically the same as the diagnostic TAPCK. TRAPCK, however runs with channel traps enabled, disabled, and inhibited. Like TAPCK, this diagnostic writes two tapes, one on each channel. Each tape then contains one file of four binary records, followed by two end-of-file marks, and a file of two BCD records, also followed by two end-of-file marks.
TRAPCK initially stores in each trap location a transfer to a routine which saves the indication of the trap location and then transfers to a subroutine TRPPED. TRPPED saves machine conditions, and then saves in the sense indicators the trap bits. If traps are supposed to be disabled or inhibited, the routine prints this and what trap occurred, and goes to restore machine conditions, and return. If a trap was expected, but a different one occurred, the routine prints both expected and occurring traps, restores machine condition, restores channel traps, and returns. If the expected trap occurred, the routine restores machine conditions, does <u>not</u> restore channel traps, and returns.

If an unexpected condition occurs, the routine places in the accumulator the location of a format statement to be used by NPRINT and enters subroutine PRTCHN, if the channel is to be printed, or PRTERR, if channel does not apply (as in I/Ø check). Both enter subroutine PRNTR, which prints an error line containing channel indication, if applicable, followed by the line specified by the format statement. The subroutine then halts. Pressing start causes TRAPCK to continue testing.

On a real tape redundancy, occurring ten consecutive times, the printed line indicates that the tape must be changed.

Subroutine TAPEX does the actual tape reading. It is entered with the read select in the accumulator, and in 1,4 the locations of the I/Ø command used to write the record and the I/Ø command to be used to read it. If the tag position of 1,4 is zero, no trap is expected. A tag of one indicates that index register one contains the trap type expected, and TAPEX will use a routine that takes approximately one second. TAPEX sets up the correct channel and tape commands and reads the record. If the delay routine is used and no trap has occurred, a printout occurs. TAPEX checks the end-of-file indicator and, if it is on, a printout occurs and return is made to the main routine. TAPEX then checks the number of words read and, if this is incorrect, a printout occurs. Each word read is checked with each word written; if any does not compare, and no redundancy was indicated, a printout occurs. If everything is correct, the redundancy light is checked, and, if it is on, a printout occurs. Return is then made to the main routine.

The main testing routine, at TPTST, initially turns off any waiting traps. At TEST1, the channel command trap is enabled, and TAPEX is entered to read the first record of the Channel A tape. Index register one contains a one, indicating that the channel command trap is expected. Upon return, traps are disabled, and the routine skips record two. An end-of-file indication causes a printout. The same test is made on the Channel B tape.

The I/Ø light is checked. With traps disabled, the routine (at TEST2) then backspaces over record two on both tapes, and enters TAPEX to read record two on both tapes. All waiting traps are then turned off.

At TEST3, the end-of-file trap is enabled, and records three and four are skipped. If a trap occurs, a subroutine prints. The routine then reads the first end-of-file, and prints if no trap occurs. With traps inhibited by the end-of-file trap (if no trap occurred, the routine disables all traps), the second end-of-file is read. The same test is made on both channels. Traps are disabled and the end-of-file indicators are turned off.

The routine, now at TEST5, reads the first record(BCD), of the second file on both tapes. At TEST6, the routine reads from both tapes the second BCD record in binary, and prints if no redundancy occurs. The routine then backspaces over the record, enables the redundancy trap, and again reads in binary. If the trap does not occur, a subroutine prints.

At TEST7, traps are disabled, add the routine backspaces on both tapes over two files, over record four of the first file and then reads record four. At TEST8, the routine backspaces a file, which should return the tape to its load point, executes two record backspaces, which should delay, and then tests the begin tape indicator, printing if it is not on. If sense switch two is down, the routine prints "ONE PASS COMPLETE" and returns to TPTST.