

CTS11-JC

CTS11-JC HOLLORITH  
CZCTBAO

AH-S076A-MC  
FICHE 1 OF 1

JUN 1980  
COPYRIGHT © 74, 80  
MADE IN USA



Microfilm frame containing a grid of frames with illegible data.



.REM \_

2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49

IDENTIFICATION

PRODUCT CODE: AC-S068A-MC  
PRODUCT NAME: CZCTBAO CTS11-JC HOLLORITH  
8035-8045 (11/70)  
80 COLUMN CARD TERMINAL CONTROL  
DATA FORMAT: 8 BIT HOLLERITH  
UPDATED DATE 20 MARCH 80  
MAINTAINER: J. BENNETT  
COMPUTER SPECIAL SYSTEMS  
AUTHOR: P.W. DUKE  
UPDATE AUTHOR: VIJAY ANANDWALA

THE INFORMATION IN THIS DOCUMENT IS SUBJECTED TO  
CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED  
AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION  
DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY  
FOR ANY ERRORS THAT MAY APPEAR IN THIS DOCUMENT.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED  
UNDER A LICENSE AND MAY ONLY BE USED OR COPIED  
IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY  
FOR THE USE OR RELIABILITY OF ITS SOFTWARE  
ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT: 1974,1980

DIGITAL EQUIPMENT CORPORATION.  
MAYNARD,MASS.

51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100  
101  
102  
103  
104  
105  
106

1.0 ABSTRACT

THIS PROGRAM TESTS THE OPERATIONAL ABILITY OF THE CTS11-JC TO INTERFACE TO AN 8035-8045 80 COLUMN READER/PUNCH. THE PROGRAM CONSISTS OF TWO SETS OF TESTS, LOGIC TESTS AND CARD TESTS. THE LOGIC TESTS ARE PERFORMED WITHOUT CARDS IN THE 8035-8045. THE CARD TESTS ARE A FOUR PASS SERIES OF TESTS WHICH USE 20 CARDS. THE CARD TESTS SHOULD NOT BE RUN UNTIL THE LOGIC TESTS HAVE RUN SUCCESSFULLY. THE PROGRAM IS DEVICE CODE INDEPENDENT AND THERE IS AN INPUT ROUTINE WHICH ALLOWS THE OPERATOR TO SPECIFY DEVICE ADDRESS, INTERRUPT VECTOR ADDRESS AND BR LEVEL. THIS ROUTINE IS RUN AT THE BEGINNING OF EVERY PASS THROUGH THE PROGRAM. THERE IS A GREAT DEAL OF INTERACTION REQUIRED BETWEEN THE OPERATOR AND THE PROGRAM. THIS INTERACTION IS ACCOMPLISHED BY MEANS OF TELETYPE DIRECTIONS AND MESSAGES TYPED OUT DURING THE RUNNING OF THE PROGRAM WHICH FREQUENTLY REQUIRES OPERATOR RESPONSE. DIRECTIONS GIVEN TO THE OPERATOR VIA THE TELETYPE MUST BE FOLLOWED EXACTLY TO ACHIEVE SUCCESSFUL TESTING.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11 WITH MINIMUM OF 4K OF MEMORY  
CTS11-JC INTERFACE  
8035-8045 80 COLUMN READER/PUNCH

2.2 PRELIMINARY PROGRAMS

ALL PROCESSOR DIAGNOSTICS MUST RUN ERROR FREE

3.0 LOADING PROCEDURE

PROCEDURE FOR LOADING NORMAL ABSOLUTE TAPES SHOULD BE FOLLOWED.

4.0 STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SR15=1 CONTINUE AFTER ERROR  
SR14=1 DELETE ERROR TYPEOUTS  
SR13=1 LOOP ON ERROR  
SR2=1 BYPASS DEVICE CODE ENTRY ROUTINE  
SR1=1 DO LOGIC TESTS  
SR0=1 DO CARD TESTS

4.2 OPERATOR ACTION



163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176  
177  
178  
179  
180  
181  
182  
183  
184  
185  
186  
187  
188  
189  
190  
191  
192  
193  
194  
195  
196  
197  
198  
199  
200  
201  
202  
203  
204  
205  
206  
207  
208  
209  
210  
211  
212  
213  
214  
215  
216  
217  
218

THE CARD TESTS ARE A SERIES OF 4 ROUTINES (PASSES) WHICH CHECK THE ABILITY OF THE INTERFACE TO PERFORM CORRECTLY IN CARD HANDLING OPERATIONS. THE TESTS REQUIRE A DECK OF 20 BLANK CARDS WHICH ARE PASSED THROUGH THE 8035-8045 4 TIMES, WHICH AT THE CONCLUSION OF THE 4 PASSES CONTAIN PUNCHED INFORMATION WHICH MUST BE VISUALLY VERIFIED BY THE OPERATOR. VERIFICATION OF CORRECT OPERATION OF THESE 4 PASSES IS DONE BY THE OPERATOR BASED ON INSTRUCTIONS GIVEN VIA TYPE OUTS AT THE BEGINNING AND END OF EACH PASS. DETAILED DESCRIPTIONS OF WHAT IS DONE DURING EACH PASS IS GIVEN BELOW.

PASS 1 - THE OPERATOR PUTS 10 BLANK CARDS IN EACH INPUT HOPPER. THE PROGRAM READS THESE CARDS AND PUTS THEM INTO THE 2 STACKERS, 10 BLANK CARDS IN EACH STACKER. THE OPERATOR VERIFIES THAT THERE ARE 10 BLANK CARDS IN EACH STACKER.

PASS 2 - THE OPERATOR PUTS THE 20 BLANK CARDS IN THE PRIMARY HOPPER. THE PROGRAM READS THESE CARDS AND ALSO PUNCHES THE CHARACTER SET ON EACH CARD. THE CARDS ARE PUNCHED IN A PRECESS FASHION WITH THE 1ST PUNCHED CHARACTER OF CARD #1 APPEARING IN COLUMN #11 AND SO ON UNTIL THE 1ST PUNCHED CHARACTER OF CARD #20 IS IN COLUMN #31. THE CARDS ARE ALL PLACED IN STACKER #1, THE OPERATOR VERIFIES THAT THE CHARACTER SET APPEARS IN EACH CARD AND THAT PUNCHING WAS DONE IN A PRECESS FASHION. THE CHARACTERS APPEAR ON THE CARDS IN THE ORDER SHOWN BELOW:

.'#\$%&'()\*+,-/0123456789:;<=>?@ABCDEFGHIJKLMN OPQRSTUVWXYZ

.REM -

PASS 3 - THE OPERATOR PUTS THE 20 CARDS IN THE PRIMARY HOPPER. THE PROGRAM READS THE CARDS AND ALSO PUNCHES MORE CHARACTERS ON EACH CARD BEGINNING WITH COLUMN #1, CARD #1 WILL HAVE THE LETTER "A" PUNCHED CARD #2 WILL HAVE 2 CHARACTERS (LETTER A) UP TO CARD #20 WHICH WILL HAVE 20 CHARACTERS (LETTER A) PUNCHED ALL THE CHARACTERS PUNCHED IN THIS TEST WILL BE CAPITAL THE CARDS WILL BE PLACED IN STACKER #1, THE OPERATOR VERIFIES THAT THESE CHARACTERS WERE PUNCHED CORRECTLY

THIS IS THE END OF THE CARD TESTS.

:CTS11-JC 80 COLUMN CARD TERMINAL CONTROL INTERFACE TEST (11/70)  
:DATA IS IN HOLERITH. (8 BIT) FORMAT  
:DECSPEC-CZCTBA  
: \*\*\*\*\* MODEL 8035-8045 \*\*\*\*\*  
:AUTHORS:  
: P. W. DUKE  
: A. L. UNSER  
:20-DEC-73

219  
220  
221  
222  
223  
224  
225 177776  
226 177570  
227 177560  
228 177562  
229 177564  
230 177566  
231 000000  
232 000001  
233 000002  
234 000003  
235 000004  
236 000005  
237 000006  
238 000007  
239 000000  
240 104000  
241 104400  
242 000004  
243 000003  
244 000002  
245 000001  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274

:UPDATE AUTHOR: VIJAY ANANDWALA  
:[UPDATED MARCH-80]

.LIST ME  
.ABS

PS=177776  
SR=177570  
TKS=177560  
TKB=177562  
TPS=177564  
TPB=177566  
R0=%0  
R1=%1  
R2=%2  
R3=%3  
R4=%4  
R5=%5  
SP=%6  
PC=%7  
XX=HALT  
ERR=EMT  
TYPE=TRAP  
COL=4  
GB=3  
G=2  
B=1

.MACR WTDONE  
BIT #200,@TCR  
BEQ .-6 ;WAIT FOR DONE FLAG  
.ENDM  
  
.MACR WTODR  
BITS #200,@TIRHI  
BEQ .-6 ;WAIT FOR OUTPUT DATA REQUEST  
.ENDM  
  
.MACR WTIDR  
BIT #200,@TIR  
BEQ .-6 ;WAIT FOR INPUT DATA REQUEST  
.ENDM  
  
.MACR LOOP A  
.NLIST  
CMP #1,SCOPE  
BNE .+16  
CMP #.-10,LASTPC  
BNE .+6  
.LIST  
JMP @NA  
.ENDM  
  
.MACR CKTCR A,B  
LP-  
MOV A,GOOD  
MOV @TCR,BAD ;READ CMD REGISTER













```

363
364 001110 013746 000004
365 001114 012737 001150 000004
366 001122 012767 177570 177550
367 001130 012767 177570 177544
368 001136 022777 177777 177534
369 001144 001012
370
371 001146 000403
372 001150 012716 001156
373 001154 000002
374 001156 012767 000176 177514
375 001164 012767 000174 177510
376 001172 012637 000004
377 001176 005067 177502
378 001202 132767 000200 177476
379 001210 001403
380 001212 012767 000710 177460
381 001220 000207
382
383
384
385
386 001222 104400
387 001224 015441
388 001226 104400
389 001230 015475
390 001232 104400
391 001234 017274
392 001236 004767 010750
393 001242 000005
394 001244
(1) 001244 001244
(1) 001244 012767 000000 017574
(1) 001252 017767 017622 017556
(1) 001260 026767 017562 017550
(1) 001266 001401
(1) 001270 104003
(1) 001272
(2) 001312 000137 001242
395 001316
(1) 001316 001316
(1) 001316 012767 000000 017522
(1) 001324 017767 017554 017504
(1) 001332 026767 017510 017476
(1) 001340 001401
(1) 001342 104003
(1) 001344
(2) 001364 000137 001242
396 001370
(1) 001370 001370
(1) 001370 012767 000000 017450
(1) 001376 017767 017512 017432
(1) 001404 026767 017436 017424
(1) 001412 001401
(1) 001414 104003
  
```

```

;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
SWRCK: MOV @#ERRVEC,-(SP) ;SAVE ERROR VECTOR
        MOV #RT1,@#ERRVEC ;SET UP ERROR VECTOR
        MOV #DSWR,SWR ;SET UP FOR HARDWARE SWR
        MC #DDISP,DISPLAY ;AND A HARDWARE DISPLAY REGISTER
        CMF #-1,@SWR ;TRY TO REFERENCE HARDWARE SWR
        BNE RT3 ;BRANCH IF NO TIME OUT TRAP OCCURED
        ;AND THE HARDWARE SWR IS NOT = -1
        BR RT2 ;BRANCH IF NO TIME OUT
RT1: MOV #RT2,(SP) ;SET UP FOR TRAP RETURN
RT2: MOV #SWREG,SWR ;POINT TO SOFTWARE SWR
        MOV #DISPREG,DISPLAY
RT3: MOV (SP)+,@#ERRVEC ;RESTORE ERROR VECTOR
        CLR APASS ;CLEAR PASS COUNT
        BITB #200,AENVM ;TEST USER SIZE UNDER APT
        BEQ RT4 ;YES,USE NON-APT SWITCH
        MOV #ASWREG,SWR ;NO,USE APT SWITCH REGISTER
RT4: RTS PC
  
```

;3 REGISTERS SHOULD EQUAL ZERO AT INITIALIZATION

```

INIT: TYPE
      LOGTST ;'***** LOGIC TESTS *****'
      TYPE
      INITM1 ;'PUT THE 8035-8045 OFF-LINE, REMOVE ALL
              TYPE
              P2MSG4 ;PRESS STOP RESET ON 8035-8045
              PC,CONTIN ;'PRESS CR TO CONTINUE'
INIT1: JSR RESET
        CKTCR #0,INIT1 ;TEST CMD REG FOR 0
        LP=.
        MOV #0,GOOD
        MOV @TCR,BAD ;READ CMD REGISTER
        CMP GOOD,BAD
        BEQ .+4
        ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPECTED
        LOOP INIT1
        JMP @#INIT1
        CKTSR #0,INIT1 ;TEST STATUS REG FOR 0
        LP=.
        MOV #0,GOOD
        MOV @TSR,BAD ;READ STATUS REGISTER
        CMP GOOD,BAD
        BEQ .+4
        ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EXPECTED
        LOOP INIT1
        JMP @#INIT1
        CKTIR #0,INIT1 ;TEST INTERRUPT REG FOR 0
        LP=.
        MOV #0,GOOD
        MOV @TIR,BAD ;READ INTERRUPT REGISTER
        CMP GOOD,BAD
        BEQ .+4
        ERR+GB ;INTERRUPT REG - VALUE READ NOT EQUAL TO EXPECTED
  
```

```

(1) 001416
(2) 001436 000137 001242
397
398 001442 000137 001446
399
400
401
402 001446 012702 001632
403 001452 012267 017372
404 001456 016777 017366 017414
405 001464
(1) 001464 001464
(1) 001464 016767 017360 017354
(1) 001472 017767 017402 017336
(1) 001500 026757 017342 017330
(1) 001506 001401
(1) 001510 104003
(1) 001512
(2) 001532 000137 001456
406 001536 020227 001664
407 001542 001343
408
409 001544 012777 074136 017326
410 001552 000005
411 001554
(1) 001554 001554
(1) 001554 012767 000000 017264
(1) 001562 017767 017312 017246
(1) 001570 026767 017252 017240
(1) 001576 001401
(1) 001600 104003
(1) 001602
(2) 001622 000137 001544
412
413 001626 000137 001664
414
415 001632 000000
416 001634 040000
417 001636 020000
418 001640 010000
419 001642 004000
420 001644 000100
421 001646 000020
422 001650 000010
423 001652 000004
424 001654 000002
425 001656 000000
426 001660 074136
427 001662 000000
428
429
430
431 001664 104400
432 001666 015653
433 001670 004767 010316
434
  
```

```

LOOP INIT1
JMP @#INIT1

JMP @#TEST1

;LOAD-READ TEST OF COMMAND REGISTER
TEST1: MOV #T1LST,R2
T1A: MOV (R2)+,HOLD
T1B: MOV HOLD,@TCR ;LOAD CMD REGISTER
      CKTCR HOLD,T1B ;TEST CMD REG FOR CORRECT VALUE
      LP=.
      MOV HOLD,GOOD
      MOV @TCR,BAD ;READ CMD REGISTER
      CMP GOOD,BAD
      BEQ .+4
      ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPECTED
      LOOP T1B
      JMP @#T1B
      CMP R2,#T1END+2
      BNE T1A

;NOW SEE IF RESET CLEARS ALL BITS THAT CAN BE SET IN THE COMMAND REGISTER
T1C: MOV #74136,@TCR ;LOAD COMMAND REG WITH ALL BITS
      RESET
      CKTCR #0,T1C ;TEST CMD REG FOR 0
      LP=.
      MOV #0,GOOD
      MOV @TCR,BAD ;READ CMD REGISTER
      CMP GOOD,BAD
      BEQ .+4
      ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPECTED
      LOOP T1C
      JMP @#T1C

JMP @#TEST2

T1LST: 0
      40000
      20000
      10000
      4000
      100
      20
      10
      4
      2
      0
      74136
T1END: 0

;TEST THAT DONE BIT IS ONLY BIT SET IN 3 REGISTERS
TEST2: TYPE
      T2MSG1 ;'PUT 8035-8045 ON-LINE AND PRESS START
      JSR PC,CONTIN ;'PRESS CR TO CONTINUE'
;DONE SHOULD BE THE ONLY BIT SET IN 3 REGISTERS
  
```

435 001674  
(1) 001674 001674  
(1) 001674 012767 000200 017144  
(1) 001702 017767 017172 017126  
(1) 001710 026767 017132 017120  
(1) 001716 001401  
(1) 001720 104003  
(1) 001722  
(2) 001742 000137 001674  
436 001746  
(1) 001746 001746  
(1) 001746 012767 000000 017072  
(1) 001754 017767 017124 017054  
(1) 001762 026767 017060 017046  
(1) 001770 001401  
(1) 001772 104003  
(1) 001774  
(2) 002014 000137 001746  
437 002020  
(1) 002020 002020  
(1) 002020 012767 000000 017020  
(1) 002026 017767 017062 017002  
(1) 002034 026767 017006 016774  
(1) 002042 001401  
(1) 002044 104003  
(1) 002046  
(2) 002066 000137 002020  
438  
439 002072 000137 002076  
440  
441  
442  
443 002076 012767 000340 175672  
444 002104 012767 000007 016744  
445 002112 012777 002204 017000  
446 002120 012777 000340 016774  
447  
448 002126 012777 000100 016744  
449 002134 162767 000040 175634  
450 002142 000240  
451 002144 000240  
452 002146 005367 016704  
453 002152 100370  
454 002154 104000  
455 002156  
(1) 002176 000137 002076  
456 002202 000427  
457  
458 002204 062706 000004  
459 002210 016767 016710 016630  
460 002216 016767 016634 016612  
461 002224 026767 016616 016604  
462 002232 001401  
463 002234 104003  
464 002236  
(1) 002256 000137 002076

```
CKTCR #200,LP ;TEST CMD REG FOR DONE FLAG
LP=
MOV #200,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPcC
LOOP LP
JMP @#LP
CKTSR #0,LP ;TEST STATUS REG FOR 0
LP=
MOV #0,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
CKTIR #0,LP ;TEST INTERRUPT REG FOR 0
LP=
MOV #0,GOOD
MOV @TIR,BAD ;READ INTERRUPT REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;INTERRUPT REG - VALUE READ NOT EQUAL TO
LOOP LP
JMP @#LP
JMP @#TEST3
;TEST THAT DONE BIT CAUSES INTERRUPT TO CORRECT ADDRESS AND BR LEVEL
TEST3: MOV #340,PS
MOV #7,LEVEL
MOV #T3INT,@PCV ;SETUP PC VECTOR
MOV #340,@PSV ;SETUP PS VECTOR
;SHOULD GET AN INTERRUPT WHEN BPU IS SET TO ONE LEVEL BELOW BR LEVEL FOR
MOV #100,@TCR ;ENABLE INTERRUPT IN CMD REG
T3A: SUB #40,PS ;DROP BPU LEVEL BY 1
NOP
NOP
DEC LEVEL ;IF LEVEL IS 0 OR HIGHER
BPL T3A ;GO WAIT FOR INT
ERR ;GOT NO INTERRUPT WITH BPU AT LEVEL 0
LOOP TEST3
JMP @#TEST3
BR T3B
;GOT INTERRUPT, NOW TEST TO SEE THAT IT OCCURRED AT CORRECT LEVEL
T3INT: ADD #4,SP ;HOUSE KEEPING OF STACK
MOV BRLV,GOOD ;GOOD=BR LEVEL OF DEVICE
MOV LEVEL,BAD ;BAD=LEVEL AT WHICH INTERRUPT OCCURRED
CMP GOOD,BAD
BEQ .+4
ERR+GB ;INTERRUPT OCCURRED AT WRONG LEVEL
LOOP TEST3
JMP @#TEST3
```

465 002262 000005  
466 002264  
(1) 002264 002264  
(1) 002264 012767 000200 016554  
(1) 002272 017767 016602 016536  
(1) 002300 026767 016542 016530  
(1) 002306 001401  
(1) 002310 104003  
(1) 002312  
(2) 002332 000137 002262  
467  
468 002336 000137 002342  
469  
470  
471  
472  
473  
474  
475 002342 112777 000040 016532  
476 002350  
(1) 002350 002350  
(1) 002350 012767 000220 016470  
(1) 002356 017767 016522 016452  
(1) 002364 026767 016456 016444  
(1) 002372 001401  
(1) 002374 104003  
(1) 002376  
(2) 002416 000137 002342  
477 002422  
(1) 002422 002422  
(1) 002422 012767 120200 016416  
(1) 002430 017767 016444 016400  
(1) 002436 026767 016404 016372  
(1) 002444 001401  
(1) 002446 104003  
(1) 002450  
(2) 002470 000137 002342  
478  
479 002474 000005  
480 002476  
(1) 002476 002476  
(1) 002476 012767 000200 016342  
(1) 002504 017767 016370 016324  
(1) 002512 026767 016330 016316  
(1) 002520 001401  
(1) 002522 104003  
(1) 002524  
(2) 002544 000137 002474  
481  
482 002550 112777 000100 016324  
483 002556  
(1) 002556 002556  
(1) 002556 012767 000220 016262  
(1) 002564 017767 016314 016244  
(1) 002572 026767 016250 016236  
(1) 002600 001401

T3B: RESET ;CLEAR OUT CMD REG WITH RESET  
CKTCR #200,T3B ;TEST CMD REG FOR DONE  
LP=.  
MOV #200,GOOD  
MOV @TCR,BAD ;READ CMD REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC  
LOOP T3B  
JMP @#T3B  
  
JMP @#TEST4  
  
;TEST THAT HOPPER EMPTY CAN BE DETECTED WHEN HOPPERS ARE SELECTED, THAT  
;HOPPER EMPTY BEING SET CAUSES INPUT ERROR TO SET AND THIS IN TURN CAUSE  
;ERROR BIT TO SET IN THE CMD REG. TEST THAT ERROR BIT CAUSES AN INTERRUPT  
;TEST FOR PRIMARY HOPPER EMPTY  
TEST4: MOV #40,@TCRHI ;SELECT PRIMARY HOPPER  
CKTSR #220,TEST4 ;TEST FOR IE + HE  
LP=.  
MOV #220,GOOD  
MOV @TSR,BAD ;READ STATUS REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX  
LOOP TEST4  
JMP @#TEST4  
CKTCR #120200,TEST4 ;TEST FOR ERR + HSO + DONE  
LP=.  
MOV #120200,GOOD  
MOV @TCR,BAD ;READ CMD REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC  
LOOP TEST4  
JMP @#TEST4  
;TEST THAT RESET WILL CLEAR OUT CMD REG  
T4A: RESET ;CLEAR OUT CMD REG WITH RESET  
CKTCR #200,T4A ;TEST CMD REG FOR DONE  
LP=.  
MOV #200,GOOD  
MOV @TCR,BAD ;READ CMD REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC  
LOOP T4A  
JMP @#T4A  
;TEST FOR SECONDARY HOPPER EMPTY  
T4B: MOV #100,@TCRHI ;SELECT SECONDARY HOPPER  
CKTSR #220,T4B ;TEST FOR IE + HE  
LP=.  
MOV #220,GOOD  
MOV @TSR,BAD ;READ STATUS REGISTER  
CMP GOOD,BAD  
BEQ .+4

(1)	002602	104003			ERR+GB		;STATUS REG - VALUE READ NOT EQUAL TO EX
(1)	002604				LOOP	T4B	
(2)	002624	000137	002550		JMP	@#T4B	
484	002630				CKTCR	#140200,T4B	;TEST FOR ERR + HS1 + DONE
(1)		002630			LP=.		
(1)	002630	012767	140200	016210	MOV	#140200,GOOD	
(1)	002636	017767	016236	016172	MOV	@TCR,BAD	;READ CMD REGISTER
(1)	002644	026767	016176	016164	CMP	GOOD,BAD	
(1)	002652	001401			BEQ	+.4	
(1)	002654	104003			ERR+GB		;CMD REG - VALUE READ NOT EQUAL TO EXPEC
(1)	002656				LOOP	T4B	
(2)	002676	000137	002550		JMP	@#T4B	
485							;TEST THAT ERROR BIT IN CMD REG CAUSES AN INT
486	002702	112777	000100	016170	MOV	#100,@TCR	;ENABLE INT
487	002710	012777	002774	016202	MOV	#T4INT,@PCV	;SETUP PC VECTOR
488	002716	012777	000340	016176	MOV	#340,@PSV	;SETUP PS VECTOR
489	002724	012767	000040	175044	MOV	#40,PS	;SET BPU TO LEVEL #1
490	002732	000240			NOP		
491	002734	000240			NOP		
492	002736	012767	000340	175032	MOV	#340,PS	;SET BPU TO LEVEL #7
493	002744	104000			ERR		;WITH ERROR BIT SET IN CMD REG INT DIDN'
494	002746				LOOP	T4C	
(1)	002766	000137	002724		JMP	@#T4C	
495	002772	000402			BR	T4D	
496	002774	062706	000004		ADD	#4,SP	;HOUSE KEEPING ON STACK
497	003000				CKTCR	#140300,T4D	;TEST FOR ERR + HS1 + IE + DONE
(1)		003000			LP=.		
(1)	003000	012767	140300	016040	MOV	#140300,GOOD	
(1)	003006	017767	016066	016022	MOV	@TCR,BAD	;READ CMD REGISTER
(1)	003014	026767	016026	016014	CMP	GOOD,BAD	
(1)	003022	001401			BEQ	+.4	
(1)	003024	104003			ERR+GB		;CMD REG - VALUE READ NOT EQUAL TO EXPEC
(1)	003026				LOOP	T4D	
(2)	003046	000137	003000		JMP	@#T4D	
498							;TEST THAT WRITING A 1 TO ERROR BIT CLEAR WILL CLEAR IT
499	003052	012777	000400	016020	MOV	#400,@TCR	;CLEAR ERROR BIT BY WRITING A 1 TO IT
500	003060				CKTCR	#200,T4E	;TEST CMD REG FOR DONE
(1)		003060			LP=.		
(1)	003060	012767	000200	015760	MOV	#200,GOOD	
(1)	003066	017767	016006	015742	MOV	@TCR,BAD	;READ CMD REGISTER
(1)	003074	026767	015746	015734	CMP	GOOD,BAD	
(1)	003102	001401			BEQ	+.4	
(1)	003104	104003			ERR+GB		;CMD REG - VALUE READ NOT EQUAL TO EXPEC
(1)	003106				LOOP	T4E	
(2)	003126	000137	003052		JMP	@#T4E	
501	003132				CKTSR	#0,T4E	;TEST STATUS REG FOR 0
(1)		003132			LP=.		
(1)	003132	012767	000000	015706	MOV	#0,GOOD	
(1)	003140	017767	015740	015670	MOV	@TSR,BAD	;READ STATUS REGISTER
(1)	003146	026767	015674	015662	CMP	GOOD,BAD	
(1)	003154	001401			BEQ	+.4	
(1)	003156	104003			ERR+GB		;STATUS REG - VALUE READ NOT EQUAL TO EX
(1)	003160				LOOP	T4E	
(2)	003200	000137	003052		JMP	@#T4E	
502							
503	003204	000137	003210		JMP	@#TEST5	



504  
505  
506  
507  
508  
509 003210 104400  
510 003212 015744  
511 003214 004767 006772  
512 003220 000005  
513 003222 112777 000010 015652  
514 003230 104400  
515 003232 015653  
516 003234 004767 006752  
517 003240 104400  
518 003242 016075  
519 003244 104400  
520 003246 016164  
521 003250 104400  
522 003252 016234  
523 003254 132777 000020 015624  
524 003262 001774  
525 003264  
(1) 003264 003264  
(1) 003264 012767 110000 015554  
(1) 003272 017767 015606 015536  
(1) 003300 026767 015542 015530  
(1) 003306 001401  
(1) 003310 104003  
(1) 003312  
(2) 003332 000137 003254  
526 003336  
(1) 003336 003336  
(1) 003336 012767 104000 015502  
(1) 003344 017767 015530 015464  
(1) 003352 026767 015470 015456  
(1) 003360 001401  
(1) 003362 104003  
(1) 003364  
(2) 003404 000137 003336  
527 003410 104400  
528 003412 017210  
529 003414 132777 000020 015464  
530 003422 001374  
531 003424  
(1) 003424 003424  
(1) 003424 012767 000000 015414  
(1) 003432 017767 015446 015376  
(1) 003440 026767 015402 015370  
(1) 003446 001401  
(1) 003450 104003  
(1) 003452  
(2) 003472 000137 003414  
532 003476  
(1) 003476 003476  
(1) 003476 012767 104000 015342  
(1) 003504 017767 015370 015324

: TEST FOR PROPER STACKER FULL DETECTION FROM BOTH STACKERS INDIVIDUALLY  
: AND THEN IN STACKER OVERFLOW MODE  
: TEST FOR STACKER #1 FULL  
TESTS: TYPE  
T5M1 ;'PUT 8035-8045 OFF-LINE'  
JSR PC,CONTIN ;'PRESS CR TO CONTINUE'  
RESET  
MOVB #10,@TCRHI ;SELECT STACKER #1  
TYPE  
T2MSG1 ;'PUT 8035-8045 ON-LINE AND PRESS START  
JSR PC,CONTIN ;'PRESS CR TO CONTINUE'  
TYPE  
T5MSG1 ;'PULL CARD PLATE OF STACKER #1 BACK TO  
TYPE  
T5MSG2 ;WILL SIMULATE STACKER FULL AND HOLD IT  
TYPE  
T5MSG3 ;'PROGRAM WILL WAIT FOR STACKER FULL FLA  
T5A: BITB #20,@TSRHI  
BEQ T5A ;WAIT FOR STACKER FULL FLAG  
CKTSR #110000,T5A ;TEST FOR OE + SF  
LP=.  
MOV #110000,GOOD  
MOV @TSR,BAD ;READ STATUS REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX  
LOOP T5A  
JMP @#T5A  
CKTCR #104000,LP ;TEST FOR ERR + SSO + DONE  
LP=.  
MOV #104000,GOOD  
MOV @TCR,BAD ;READ CMD REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC  
LOOP LP  
JMP @#LP  
TYPE  
P2MSG3 ;' PRESS STOP-RESET AND PRESS START ON T  
T5B: BITB #20,@TSRHI  
BNE T5B ;WAIT FOR STACKER FULL TO GO AWAY  
CKTSR #0,T5B ;TEST STATUS REG FOR 0  
LP=.  
MOV #0,GOOD  
MOV @TSR,BAD ;READ STATUS REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX  
LOOP T5B  
JMP @#T5B  
CKTCR #104000,LP ;TEST FOR ERR + SSO  
LP=.  
MOV #104000,GOOD  
MOV @TCR,BAD ;READ CMD REGISTER

(1)	003512	026767	015330	015316			CMP	GOOD,BAD	
(1)	003520	001401					BEQ	.+4	
(1)	003522	104003					ERR+GB		;CMD REG - VALUE READ NOT EQUAL TO EXPEC
(1)	003524						LOOP	LP	
(2)	003544	000137	003476				JMP	@#LP	
533									;TEST FOR STACKER #2 FULL
534	003550	112777	000031	015324			MOVB	#31,@TCRHI	;SELECT STACKER #2 + CLEAR ERR
535	003556	104400					TYPE		
536	003560	016344					T5MSG5		;'PULL CARD PLATE OF STACKER #2 BACK TO
537	003562	104400					TYPE		
538	003564	016164					T5MSG2		;WILL SIMULATE STACKER FULL AND HOLD IT
539	003566	104400					TYPE		
540	003570	016234					T5MSG3		;'PROGRAM WILL WAIT FOR STACKER FULL FLA
541	003572	132777	000020	015306	T5C:		BITB	#20,@TSRHI	
542	003600	001774					BEQ	T5C	;WAIT FOR STACKER FULL FLAG
543	003602						CKTSR	#110000,T5C	;TEST FOR OE + SF
(1)		003602					LP=.		
(1)	003602	012767	110000	015236			MOV	#110000,GOOD	
(1)	003610	017767	015270	015220			MOV	@TSR,BAD	;READ STATUS REGISTER
(1)	003616	026767	015224	015212			CMP	GOOD,BAD	
(1)	003624	001401					BEQ	.+4	
(1)	003626	104003					ERR+GB		;STATUS REG - VALUE READ NOT EQUAL TO EX
(1)	003630						LOOP	T5C	
(2)	003650	000137	003572				JMP	@#T5C	
544	003654	104400					TYPE		
545	003656	017210					P2MSG3		;'PRESS STOP-RESET AND PRESS START ON THE 8035-8
546	003660	132777	000020	015220	T5D:		BITB	#20,@TSRHI	
547	003666	001374					BNE	T5D	;WAIT FOR STACKER FULL TO GO AWAY
548	003670						CKTSR	#0,T5D	;TEST STATUS REG FOR 0
(1)		003670					LP=.		
(1)	003670	012767	000000	015150			MOV	#0,GOOD	
(1)	003676	017767	015202	015132			MOV	@TSR,BAD	;READ STATUS REGISTER
(1)	003704	026767	015136	015124			CMP	GOOD,BAD	
(1)	003712	001401					BEQ	.+4	
(1)	003714	104003					ERR+GB		;STATUS REG - VALUE READ NOT EQUAL TO EX
(1)	003716						LOOP	T5D	
(2)	003736	000137	003660				JMP	@#T5D	
549									;TEST FOR STACKER FULL IN STACKER OVERFLOW MODE
550	003742	104400					TYPE		
551	003744	016036					T5M2		;'PRESS START ON THE 8035-8045'
552	003746	004767	006240				JSR	PC,CONTIN	;'PRESS CR TO CONTINUE
553	003752	000005					RESET		;CLEAR OUT CMD REG
554	003754	012777	000001	015116			MOV	#1,@TCR	;SET STACKER OVERFLOW MODE + GO
555	003762	104400					TYPE		
556	003764	016075					T5MSG1		;'PULL CARD PLATE OF STACKER #1 BACK TO
557	003766	104400					TYPE		
558	003770	016164					T5MSG2		;WILL SIMULATE STACKER FULL AND RELEASE''
559	003772	004767	006214				JSR	PC,CONTIN	;'PRESS CR TO CONTINUE''
560	003776	132777	000020	015102	T5E:		BITB	#20,@TSRHI	
561	004004	001401					BEQ	.+4	
562	004006	104000					ERR		;STACKER FULL FLAG SET WITH ONLY STACKER
563									;FULL WHEN IN OVERFLOW MODE
564	004010						LOOP	T5E	
(1)	004030	000137	003776				JMP	@#T5E	
565	004034	104400					TYPE		
566	004036	016344					T5MSG5		;'PULL CARD PLATE OF STACKER #2 BACK TO

567 004040 104400  
568 004042 016164  
569 004044 104400  
570 004046 016234  
571 004050 132777 000020 015030  
572 004056 001774  
573 004060  
(1) 004060 004060  
(1) 004066 012767 110000 014760  
(1) 004074 026767 014746 014734  
(1) 004102 001401  
(1) 004104 104003  
(1) 004106  
(2) 004126 000137 004050  
574 004132  
(1) 004132 004132  
(1) 004132 012767 100000 014706  
(1) 004140 017767 014734 014670  
(1) 004146 026767 014674 014662  
(1) 004154 001401  
(1) 004156 104003  
(1) 004160  
(2) 004200 000137 004132  
575 004204 104400  
576 004206 016472  
577 004210 004767 005776  
578 004214 000005  
579  
580 004216 000137 004222  
581  
582  
583  
584 004222 000005  
585 004224 104400  
586 004226 016531  
587 004230 004767 005756  
588 004234 012777 020001 014636  
589 004242 005067 014574  
590 004246 162767 000001 014566  
591 004254 001374  
592 004256 162767 000001 014556  
593 004264 001374  
594 004266 162767 000001 014546  
595 004274 001374  
596 004276 162767 000001 014536  
597 004304 001374  
598 004306  
(1) 004306 004306  
(1) 004306 012767 000320 014532  
(1) 004314 017767 014564 014514  
(1) 004322 026767 014520 014506  
(1) 004330 001401  
(1) 004332 104003  
(1) 004334  
(2) 004354 000137 004306

TYPE  
T5MSG2 ;WILL SIMULATE STACKER FULL AND HOLD IT  
TYPE  
T5MSG3 ;'PROGRAM WILL WAIT FOR STACKER FULL FLA  
TSF: BITB #20,@TSRHI  
BEQ TSF ;WAIT FOR STACKER FULL FLAG  
CKTSR #110000,TSF ;TEST FOR OE + SF  
LP=.  
MOV #110000,GOOD  
MOV @TSR,BAD ;READ STATUS REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX  
LOOP TSF  
JMP @#TSF  
CKTCR #100000,LP ;TEST FOR ERR  
LP=.  
MOV #100000,GOOD  
MOV @TCR,BAD ;READ CMD REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC  
LOOP LP  
JMP @#LP  
TYPE  
T5MSG7 ;'RELEASE BOTH CARD PLATES AND PRESS STA  
JSR PC,CONTIN ;'PRESS CR TO CONTINUE'  
RESET  
JMP @#TEST6  
;FORCE INPUT CHECK ERROR BY TRYING TO FEED A CARD FROM AN EMPTY HOPPER  
TEST6: RESET  
TYPE  
T6MSG1 ;'PRESS START ON THE 8035-8045'  
JSR PC,CONTIN ;'PRESS CR TO CONTINUE'  
MOV #20001,@TCR ;ISSUE GO COMMAND TO PRIMARY HOPPER  
CLR CNT  
SUB #1,CNT  
BNE .-6  
SUB #1,CNT  
BNE .-6 ;DELAY 1 SECOND  
SUB #1,CNT  
BNE .-6  
SUB #1,CNT  
BNE .-6 ;DELAY 1 SECOND  
CKTSR #320,LP ;TEST FOR IE + IC + ME  
LP=.  
MOV #320,GOOD  
MOV @TSR,BAD ;READ STATUS REGISTER  
CMP GOOD,BAD  
BEQ .+4  
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX  
LOOP LP  
JMP @#LP













(1)	006452	017767	012426	012356
(1)	006460	026767	012362	012350
(1)	006466	001401		
(1)	006470	104003		
(1)	006472			
(2)	006512	000137	006444	
714	006516			
(1)		006516		
(1)	006516	012767	020200	012322
(1)	006524	017767	012350	012304
(1)	006532	026767	012310	012276
(1)	006540	001401		
(1)	006542	104003		
(1)	006544			
(2)	006564	000137	006516	
715				
716	006570	012767	020104	012270
717	006576	012767	000022	012240
718	006604	012777	020027	012266
719	006612			
(1)	006612	132777	000200	012276
(1)	006620	001774		
720	006622			
(1)		006622		
(1)	006622	012767	004410	012216
(1)	006630	017767	012250	012200
(1)	006636	026767	012204	012172
(1)	006644	001401		
(1)	006646	104003		
(1)	006650			
(2)	006670	000137	006622	
721	006674	004767	003534	
722	006700	162767	000002	012160
723	006706			
(1)	006706	032777	000200	012164
(1)	006714	001774		
724	006716			
(1)		006716		
(1)	006716	012767	004010	012122
(1)	006724	017767	012154	012104
(1)	006732	026767	012110	012076
(1)	006740	001401		
(1)	006742	104003		
(1)	006744			
(2)	006764	000137	006716	
725	006770			
(1)		006770		
(1)	006770	012767	020226	012050
(1)	006776	017767	012076	012032
(1)	007004	026767	012036	012024
(1)	007012	001401		
(1)	007014	104003		
(1)	007016			
(2)	007036	000137	006770	
726	007042			
(1)		007042		

```
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
CKTCR #20200,LP ;TEST FOR HSO + DONE
LP=.
MOV #20200,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP
;FEED CARDS #2-19, INHIBIT READ FOR THESE CARDS
MOV #BF1,PTR
MOV #22,CNT1 ;CNT1=18
P2A: MOV #20027,@TCR ;ISSUE HSO + IR + PRI + PUN + GO
WTO DR
BITB #200,@TIRHI
BEQ .-6 ;WAIT FOR OUTPUT DATA REQUEST
CKTSR #4410,LP ;TEST FOR PUDR
LP=.
MOV #4410,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
JSR PC,P2SEND ;SEND DATA TO PUNCH
SUB #2,PTR
WTDONE
BIT #200,@TCR
BEQ .-6 ;WAIT FOR DONE FLAG
CKTSR #4010,LP ;TEST FOR CIW
LP=.
MOV #4010,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
CKTCR #20226,LP ;TEST FOR HSO + DONE + IR + PRI + PUN
LP=.
MOV #20226,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP
CKTIR #0,LP ;TEST FOR 0
LP=.
```



(1)		007412		
(1)	007412	012767	000006	011426
(1)	007420	017767	011454	011410
(1)	007426	026767	011414	011402
(1)	007434	001401		
(1)	007436	104003		
(1)	007440			
(2)	007460	000137	007412	
744				
745	007464	152777	000100	011406
746	007472	012777	007556	011420
747	007500	012777	000340	011414
748	007506	012767	000040	170262
749	007514	000240		
750	007516	000240		
751	007520	012767	000340	170250
752	007526	104000		
753	007530			
(1)	007550	000137	007526	
754	007554	000402		
755	007556	062706	000004	
756	007562	004767	002646	
757	007566			
(1)	007566	032777	000200	011304
(1)	007574	001774		
758	007576			
(1)		007576		
(1)	007576	012767	000000	011242
(1)	007604	017767	011274	011224
(1)	007612	026767	011230	011216
(1)	007620	001401		
(1)	007622	104003		
(1)	007624			
(2)	007644	000137	007576	
759	007650			
(1)		007650		
(1)	007650	012767	000306	011170
(1)	007656	017767	011216	011152
(1)	007664	026767	011156	011144
(1)	007672	001401		
(1)	007674	104003		
(1)	007676			
(2)	007716	000137	007650	
760				
761	007722	000137	007726	
762				
763				
764				
765				
766				
767				
768	007726	104400		
769	007730	017334		
770	007732	104400		
771	007734	017422		
772	007736	104400		

```

LP=.
MOV #6,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP
;SEE IF OUTPUT DATA REQUEST WILL CAUSE AN INTERRUPT
BISB #100,@TCR ;ENABLE INTERRUPT
MOV #P2INT,@PCV ;SETUP PC VECTOR
MOV #340,@PSV ;SETUP PS VECTOR
MOV #40,PS ;SET BPU TO LEVEL #1
NOP
NOP
MOV #340,PS ;SET BPU TO LEVEL #7
P2B: ERR ;OUTPUT DATA REQUEST DIDN'T CAUSE AN INT
LOOP P2B
JMP @#P2B
BR P2C
P2INT: ADD #4,SP ;HOUSE KEEPING ON STACK
P2C: JSR PC,P2SEND ;SEND DATA TO PUNCH
WTDONE
BIT #200,@TCR
BEQ .-6 ;WAIT FOR DONE FLAG
CKTSR #0,LP ;TEST FOR STATUS =0
LP=.
MOV #0,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
CKTCR #306,LP? ;TEST FOR DONE + INT EN + PRI + PUN
LP=.
MOV #306,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP
JMP @#PASS3
;THIS TEST PERFORMS PUNCHING WITHOUT PRINTING. ONE ALL 'A'S LETTER IS
;PUNCHED IN COLUMN 1 OF CARD 1 AND SO ON UNTIL 20 ALL 'A'S LETTER
;ARE PUNCHED IN THE 1ST 20 COLUMNS OF CARD 20. THIS TEST
;VERIFIES THE DATA READ FROM PUNCHING DONE IN PASS 2.
PASS3: TYPE
P3MSG1 ;'VERIFY THAT ALL CHARS WERE PUNCHED ON
TYPE
P3MSG2 ;PRECESS FASHION BEGINNING WITH COLUMN 1
TYPE

```

773	007740	017136		
774	007742	104400		
775	007744	017210		
776	007746	004767	002240	
777				
778	007752	012767	020104	011072
779	007760	012777	020401	011112
780	007766			
(1)	007766	032777	000200	011120
(1)	007774	001774		
781	007776	004767	002776	
782	010002	162767	000002	011042
783	010010			
(1)	010010	032777	000200	011062
(1)	010016	001774		
784	010020			
(1)		010020		
(1)	010020	012767	004010	011020
(1)	010026	017767	011052	011002
(1)	010034	026767	011006	010774
(1)	010042	001401		
(1)	010044	104003		
(1)	010046			
(2)	010066	000137	010020	
785	010072			
(1)		010072		
(1)	010072	012767	020200	010746
(1)	010100	017767	010774	010730
(1)	010106	026767	010734	010722
(1)	010114	001401		
(1)	010116	104003		
(1)	010120			
(2)	010140	000137	010072	
786				
787				
788	010144	012767	000022	010672
789	010152	012767	000001	010660
790	010160	012777	020003	010712
791	010166			
(1)	010166	132777	000200	010722
(1)	010174	001774		
792	010176	004767	002306	
793	010202	062767	000001	010630
794	010210			
(1)	010210	032777	000200	010676
(1)	010216	001774		
795	010220	004767	002554	
796	010224	162767	000002	010620
797	010232			
(1)	010232	032777	000200	010640
(1)	010240	001774		
798	010242			
(1)		010242		
(1)	010242	012767	020202	010576
(1)	010250	017767	010624	010560
(1)	010256	026767	010564	010552

```

P2MSG2          : 'PLACE ALL 20 CARDS IN THE PRIMARY HOPP
TYPE
P2MSG3          : 'PRESS STOP-RESET AND PRESS START ON T
JSR             : 'PRESS CR TO CONTINUE'
PC,CONTIN
;FEED CARD #1
MOV             #BF1,INPTR
MOV             #20401,@TCR          :ISSUE CLEAR ERR + HSO + GO
WTIDR
BIT             #200,@TIR
BEQ             #-6                  :WAIT FOR INPUT DATA REQUEST
JSR             PC,CKDATA           :CHECK INPUT DATA FROM PASS 2
SUB             #2,INPTR
WTDONE
BIT             #200,@TCR
BEQ             #-6                  :WAIT FOR DONE FLAG
CKTSR           #4010,LP            :TEST FOR CIW
LP=.
MOV             #4010,GOOD
MOV             @TSR,BAD             :READ STATUS REGISTER
CMP             GOOD,BAD
BEQ             .+4
ERR+GB
LOOP           LP
JMP             @NLP
CKTCR           #20200,LP          :TEST FOR HSO + DONE
LP=.
MOV             #20200,GOOD
MOV             @TCR,BAD             :READ CMD REGISTER
CMP             GOOD,BAD
BEQ             .+4
ERR+GB
LOOP           LP
JMP             @NLP
;FEED CARDS #2-19, CHECKING INPUT DATA AND PUNCHING 1 CHAR IN 1ST CARD,
; 2 IN 2ND ETC. UP TO 20 CHARS IN CARD 20
MOV             #22,CNT1             :CNT1=18
MOV             #1,CARD              :CARD=1
P3A: MOV         #20003,@TCR         :ISSUE HSO + PUN + GO
WTODR
BITB            #200,@TIRHI
BEQ             #-6                  :WAIT FOR OUTPUT DATA REQUEST
JSR             PC,P3SEND           :SEND DATA TO PUNCH
ADD             #1,CARD
WTIDR
BIT             #200,@TIR
BEQ             #-6                  :WAIT FOR INPUT DATA REQUEST
JSR             PC,CKDATA           :CHECK INPUT DATA FROM PASS 2
SUB             #2,INPTR
WTDONE
BIT             #200,@TCR
BEQ             #-6                  :WAIT FOR DONE FLAG
CKTCR           #20202,LP          :TEST FOR HSO + DONE + PUN
LP=.
MOV             #20202,GOOD
MOV             @TCR,BAD             :READ CMD REGISTER
CMP             GOOD,BAD
  
```

(1) 010264 001401  
(1) 010266 104003  
(1) 010270  
(2) 010310 000137 010242  
799 010314  
(1) 010314 010314  
(1) 010314 012767 004010 010524  
(1) 010322 017767 010556 010506  
(1) 010330 026767 010512 010500  
(1) 010336 001401  
(1) 010340 104003  
(1) 010342  
(2) 010362 000137 010314  
800 010366  
(1) 010366 010366  
(1) 010366 012767 000000 010452  
(1) 010374 017767 010514 010434  
(1) 010402 026767 010440 010426  
(1) 010410 001401  
(1) 010412 104003  
(1) 010414  
(2) 010434 000137 010366  
801 010440 005367 010400  
802 010444 001245  
803  
804 010446 012777 020003 010424  
805 010454  
(1) 010454 132777 000200 010434  
(1) 010462 001774  
806 010464 004767 002020  
807 010470 062767 000001 010342  
808 010476  
(1) 010476 032777 000200 010410  
(1) 010504 001774  
809 010506 004767 002266  
810 010512  
(1) 010512 032777 000200 010360  
(1) 010520 001774  
811 010522  
(1) 010522 010522  
(1) 010522 012767 004230 010316  
(1) 010530 017767 010350 010300  
(1) 010536 026767 010304 010272  
(1) 010544 001401  
(1) 010546 104003  
(1) 010550  
(2) 010570 000137 010522  
812 010574  
(1) 010574 010574  
(1) 010574 012767 120202 010244  
(1) 010602 017767 010272 010226  
(1) 010610 026767 010232 010220  
(1) 010616 001401  
(1) 010620 104003  
(1) 010622  
(2) 010642 000137 010574

```
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
CKTSR    #4010,LP
LP=.
MOV      #4010,GOOD
MOV      @TSR,BAD
CMP      GOOD,BAD
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
CKTIR    #0,LP
LP=.
MOV      #0,GOOD
MOV      @TIR,BAD
CMP      GOOD,BAD
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
DEC      CNT1
BNE
;FEED CARD #20
MOV      #20003,@TCR
WTDOR
BITB     #200,@TIRHI
BEQ      -6
JSR      PC,P3SEND
ADD      #1,CARD
WTIDR
BIT      #200,@TIR
BEQ      -6
JSR      PC,CKDATA
WTDONE
BIT      #200,@TCR
BEQ      -6
CKTSR    #4230,LP
LP=.
MOV      #4230,GOOD
MOV      @TSR,BAD
CMP      GOOD,BAD
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
CKTCR    #120202,LP
LP=.
MOV      #120202,GOOD
MOV      @TCR,BAD
CMP      GOOD,BAD
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
```

;CMD REG - VALUE READ NOT EQUAL TO EXPEC  
;TEST FOR CIW  
;READ STATUS REGISTER  
;STATUS REG - VALUE READ NOT EQUAL TO EX  
;TEST INTERRUPT REG FOR 0  
;READ INTERRUPT REGISTER  
;INTERRUPT REG - VALUE READ NOT EQUAL TO  
;GO PUNCH AND READ MORE CARDS  
;ISSUE HSO + PUN + GO  
;WAIT FOR OUTPUT DATA REQUEST  
;SEND DATA TO PUNCH  
;WAIT FOR INPUT DATA REQUEST  
;CHECK INPUT DATA FROM PASS 2  
;WAIT FOR DONE FLAG  
;TEST FOR CIW + IE + HE  
;READ STATUS REGISTER  
;STATUS REG - VALUE READ NOT EQUAL TO EX  
;TEST FOR ERR + HSO + PUN + DONE  
;READ CMD REGISTER  
;CMD REG - VALUE READ NOT EQUAL TO EXPEC

813  
814 010646 012777 000003 010224  
815 010654  
(1) 010654 132777 000200 010234  
(1) 010662 001774  
816 010664 004767 001620  
817 010670  
(1) 010670 032777 000200 010202  
(1) 010676 001774  
818 010700  
(1) 010700 010700  
(1) 010700 012767 000000 010140  
(1) 010706 017767 010172 010122  
(1) 010714 026767 010126 010114  
(1) 010722 001401  
(1) 010724 104003  
(1) 010726  
(2) 010746 000137 010700  
819 010752  
(1) 010752 010752  
(1) 010752 012767 000202 010066  
(1) 010760 017767 010114 010050  
(1) 010766 026767 010054 010042  
(1) 010774 001401  
(1) 010776 104003  
(1) 011000  
(2) 011020 000137 010752  
820 011024 000137 011030  
821  
822  
823  
824  
825  
826  
827 011030 104400  
828 011032 017510  
829  
830 011034 104400  
831 011036 017577  
832 011040 104400  
833 011042 017645  
834 011044 104400  
835 011046 017136  
836 011050 104400  
837 011052 017210  
838 011054 004767 001132  
839 011060 004767 001330  
840  
841  
842 011064 012767 020104 007760  
843 011072 012777 020401 010000  
844 011100  
(1) 011100 032777 000200 010006  
(1) 011106 001774  
845 011110 004767 001664  
846 011114 162767 000002 007730

```
;PUNCH CARD #20
MOV #3,@TCR ;ISSUE PUN + GO
WTODR
BITB #200,@TIRHI
BEQ #-6 ;WAIT FOR OUTPUT DATA REQUEST
JSR PC,P3SEND ;SEND DATA TO PUNCH
WTDONE
BIT #200,@TCR
BEQ #-6 ;WAIT FOR DONE FLAG
CKTSR #0,LP ;TEST FOR STATUS=0
LP=.
MOV #0,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
CKTCR #000202,LP ;TEST FOR ERR + DONE
LP=.
MOV #000202,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP
JMP @#PASS4

; THIS TEST PERFORMS A SEPARATE PRINT OPERATION ON ALL CARDS BY
; PRINTING "CARD 01" IN COLUMNS 1-7 OF CARD 1 UP TO "CARD 20" IN
; COLUMNS 1-7 OF CARD 20. VERIFY CORRECT DATA READ FROM PUNCHING
; DONE IN PASSES 2 + 3
PASS4: TYPE
P4MSG1 ;"VERIFY THAT THERE IS THE LETTER "A"
; PUNCHED IN CARD #1
TYPE
P4MSG2 ;THAT IS HOLE PUNCHED IN ROW 12 AND 1
TYPE
P4MSG3 ;UP TO 20 ALL A'S CHARS PUNCHED IN CARD
TYPE
P2MSG2 ;"PLACE ALL 20 CARDS IN THE PRIMARY HOPP
TYPE
P2MSG3 ;"PRESS STOP-RESET AND PRESS START ON T
; "PRESS CR TO CONTINUE"
JSR PC,CONTIN ;PREPARE BUFFER FOR DATA COMPARE
JSR PC,SETBFS

;FEED CARD #1
.EVEN
MOV #BF1,INPTR
MOV #20401,@TCR ;ISSUE CLEAR ERR + HSO + GO
WTIDR
BIT #200,@TIR
BEQ #-6 ;WAIT FOR INPUT DATA REQUEST
JSR PC,CKDATA ;CHECK INPUT DATA FROM PASSES 2 + 3
SUB #2,INPTR
```

847	011122			
(1)	011122	032777	000200	007750
(1)	011130	001774		
848	011132			
(1)		011132		
(1)	011132	012767	004010	007706
(1)	011140	017767	007740	007670
(1)	011146	026767	007674	007662
(1)	011154	001401		
(1)	011156	104003		
(1)	011160			
(2)	011200	000137	011132	
849	011204			
(1)		011204		
(1)	011204	012767	020200	007634
(1)	011212	017767	007662	007616
(1)	011220	026767	007622	007610
(1)	011226	001401		
(1)	011230	104003		
(1)	011232			
(2)	011252	000137	011204	
850				
851	011256	012767	000022	007560
852	011264	012767	000001	007546
853	011272	012767	020404	007562
854	011300	012777	020017	007572
855	011306			
(1)	011306	132777	000200	007602
(1)	011314	001774		
856	011316	004767	001262	
857	011322	004767	001346	
858	011326	062767	000001	007504
859	011334			
(1)	011334	032777	000200	007552
(1)	011342	001774		
860	011344	032777	000200	007542
861	011352	001774		
862	011354	004767	001420	
863	011360	162767	000002	007464
864	011366			
(1)	011366	032777	000200	007504
(1)	011374	001774		
865	011376			
(1)		011376		
(1)	011376	012767	020216	007442
(1)	011404	017767	007470	007424
(1)	011412	026767	007430	007416
(1)	011420	001401		
(1)	011422	104003		
(1)	011424			
(2)	011444	000137	011376	
866	011450			
(1)		011450		
(1)	011450	012767	004010	007370
(1)	011456	017767	007422	007352
(1)	011464	026767	007356	007344

```

WTDONE
BIT #200,@TCR
BEQ #-6 ;WAIT FOR DONE FLAG
CKTSR #4010,LP ;TEST FOR CIW
LP=.
MOV #4010,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
CKTCR #20200,LP ;TEST FOR HSO + DONE
LP=.
MOV #20200,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP
;FEED CARDS #2-19, CHECKING INPUT DATA AND PRINTING IN COLUMNS 1-7
MOV #22,CNT1 ;CNT1=18
MOV #1,CARD ;CARD=1
MOV #CARD01,OUTPTR ;OUTPTR=CARD01
P4A: MOV #20017,@TCR ;ISSUE HSO + SPD + PRI + PUN + GO
WTDOR
BITB #200,@TIRHI
BEQ #-6 ;WAIT FOR OUTPUT DATA REQUEST
JSR PC,P5SEND ;SEND DATA TO PUNCH
JSR PC,P4SEND ;SEND DATA TO PRINT
ADD #1,CARD
WTDOR
BIT #200,@TIR
BEQ #-6 ;WAIT FOR INPUT DATA REQUEST
BIT #200,@TIR
BEQ #-6 ;WAIT FOR DONE FLAG
JSR PC,CKDATA ;CHECK INPUT DATA FROM PASSES 2 + 3
SUB #2,INPTR
WTDONE
BIT #200,@TCR
BEQ #-6 ;WAIT FOR DONE FLAG
CKTCR #20216,LP ;TEST FOR HSO + DONE + SPD + PRI + PUN
LP=.
MOV #20216,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP
CKTSR #4010,LP ;TEST FOR CIW
LP=.
MOV #4010,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
  
```

(1)	011472	001401		
(1)	011474	104003		
(1)	011476			
(2)	011516	000137	011450	
867	011522			
(1)		011522		
(1)	011522	012767	000000	007316
(1)	011530	017767	007360	007300
(1)	011536	026767	007304	007272
(1)	011544	001401		
(1)	011546	104003		
(1)	011550			
(2)	011570	000137	011522	
868	011574	005367	007244	
869	011600	001237		
870				
871	011602	012777	020017	007270
872	011610			
(1)	011610	132777	000200	007300
(1)	011616	001774		
873	011620	132777	000200	007270
874	011626	001774		
875	011630	062767	000001	007202
876	011636	004767	000742	
877	011642	004767	001026	
878	011646			
(1)	011646	032777	000200	007240
(1)	011654	001774		
879	011656	004767	001116	
880	011662	162767	000002	007162
881	011670			
(1)	011670	032777	000200	007202
(1)	011676	001774		
882	011700			
(1)		011700		
(1)	011700	012767	004230	007140
(1)	011706	017767	007172	007122
(1)	011714	026767	007126	007114
(1)	011722	001401		
(1)	011724	104003		
(1)	011726			
(2)	011746	000137	011700	
883	011752			
(1)		011752		
(1)	011752	012767	120216	007066
(1)	011760	017767	007114	007050
(1)	011766	026767	007054	007042
(1)	011774	001401		
(1)	011776	104003		
(1)	012000			
(2)	012020	000137	011752	
884				
885	012024	012777	000417	007046
886	012032			
(1)	012032	132777	000200	007056
(1)	012040	001774		

```

BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
CKTIR   #0,LP
LP=.
MOV      #0,GOOD
MOV      @TIR,BAD
CMP      GOOD,BAD
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
DEC      CNT1
BNE      P4A
;FEED CARD #20
MOV      #20017,@TCR
WTODR
BITB     #200,@TIRHI
BEQ      .-6
BITB     #200,@TIRHI
BEQ      .-6
ADD      #1,CARD
JSR      PC,P5SEND
JSR      PC,P4SEND
WTIDR
BIT      #200,@TIR
BEQ      .-6
JSR      PC,CKDATA
SUB      #2,INPTR
          WTDONE
BIT      #200,@TCR
BEQ      .-6
CKTSR   #4230,LP
LP=.
MOV      #4230,GOOD
MOV      @TSR,BAD
CMP      GOOD,BAD
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
CKTCR   #120216,LP
LP=.
MOV      #120216,GOOD
MOV      @TCR,BAD
CMP      GOOD,BAD
BEQ      .+4
ERR+GB
LOOP     LP
JMP      @#LP
;PRINT CARD #20
MOV      #000417,@TCR
WTODR
BITB     #200,@TIRHI
BEQ      .-6

```

;STATUS REG - VALUE READ NOT EQUAL TO EX  
 ;TEST INTERRUPT REG FOR 0  
 ;READ INTERRUPT REGISTER  
 ;INTERRUPT REG - VALUE READ NOT EQUAL TO  
 ;GO PRINT AND READ MORE CARDS  
 ;ISSUE HSO + SPD + PRI + PUN + GO  
 ;WAIT FOR OUTPUT DATA REQUEST  
 ;WAIT FOR OUTPUT DATA REQUEST  
 ;SEND DATA TO PUNCH  
 ;SEND DATA TO PRINTER  
 ;WAIT FOR INPUT DATA REQUEST  
 ;CHECK INPUT DATA FROM PASSES 2 + 3  
 ;WAIT FOR DONE FLAG  
 ;TEST FOR CIW + IE + ME  
 ;READ STATUS REGISTER  
 ;STATUS REG - VALUE READ NOT EQUAL TO EX  
 ;TEST FOR ERR + HSO + SPD + PRI + PUN +  
 ;READ CMD REGISTER  
 ;CMD REG - VALUE READ NOT EQUAL TO EXPEC  
 ;ISSUE CLEAR ERR + SPD + PRI + PUN + GO  
 ;WAIT FOR OUTPUT DATA REQUEST



887	012042	004767	000536	
888	012046	004767	000622	
889	012052			
(1)	012052	032777	000200	007020
(1)	012060	001774		
890	012062			
(1)		012062		
(1)	012062	012767	000000	006756
(1)	012070	017767	007010	006740
(1)	012076	026767	006744	006732
(1)	012104	001401		
(1)	012106	104003		
(1)	012110			
(2)	012130	000137	012062	
891	012134			
(1)		012134		
(1)	012134	012767	000216	006704
(1)	012142	017767	006732	006666
(1)	012150	026767	006672	006660
(1)	012156	001401		
(1)	012160	104003		
(1)	012162			
(2)	012202	000137	012134	
892				
893	012206	000005		
894	012210	000207		
895				
896				
897				
898	012212	104400		
899	012214	015550		
900	012216	004767	001734	
901	012222	004767	002642	
902	012226	022704	000015	
903	012232	001373		
904	012234	004767	002554	
905	012240	000207		
906				
907				
908				
909	012242	012700	000001	
910	012246	004767	001704	
911	012252	032777	000200	006634
912	012260	001774		
913	012262	012767	000000	006556
914	012270	017767	006614	006540
915	012276	026767	006544	006532
916	012304	001401		
917	012306	104007		
918	012310	005200		
919	012312	022709	000121	
920	012316	001355		
921	012320			
(1)		012320		
(1)	012320	012767	000000	006520
(1)	012326	017767	006562	006502

```

JSR PC,P5SEND ;SEND DATA TO PUNCH
JSR PC,P4SEND ;SEND DATA TO PRINTER
WTDONE
BIT #200,@TCR
BEQ #-6 ;WAIT FOR DONE FLAG
CKTSR #0,LP ;TEST FOR STATUS=0
LP=.
MOV #0,GOOD
MOV @TSR,BAD ;READ STATUS REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;STATUS REG - VALUE READ NOT EQUAL TO EX
LOOP LP
JMP @#LP
CKTCR #216,LP ;TEST FOR DONE + SPD + PRI + PUN
LP=.
MOV #216,GOOD
MOV @TCR,BAD ;READ CMD REGISTER
CMP GOOD,BAD
BEQ .+4
ERR+GB ;CMD REG - VALUE READ NOT EQUAL TO EXPEC
LOOP LP
JMP @#LP

RESET
RTS PC ;EXIT CARD TEST

;SUBROUTINE TO WAIT FOR CR
CONTIN: TYPE
CONMSG ;'PRESS CR TO CONTINUE'
CONT1: JSR PC,CKSWR
CMP #15,R4 ;GET A CHAR
BNE CONT1 ;WAS IT CR ?
JSR PC,TTYOUT ;NO - KEEP WAITING
RTS PC ;YES - TYPE CR + LF

;SUBROUTINE TO TEST FOR BLANK CARD (ALL SPACES)
BLKCRD: MOV #1,R0
BC1: JSR PC,CKSWR
BIT #200,@TIR ;WAIT FOR INPUT DATA REQUEST
BEQ BC1 ;READ DATA REGISTER
MOV #0,GOOD
MOV @TDR,BAD
CMP GOOD,BAD
BEQ .+4
ERR+GB+COL ;DATA REG - COMPARE ERROR AT INDICATED C
INC R0
CMP #121,R0 ;IF MORE COLUMNS TO READ GO DO IT
BNE BC1 ;TEST THAT IDR IS 0
CKTIR #0,LP
LP=.
MOV #0,GOOD
MOV @TIR,BAD ;READ INTERRUPT REGISTER
  
```

MA  
CZC

1

1

1

1

1

1

1

(1)	012334	026767	006506	006474
(1)	012342	001401		
(1)	012344	104003		
(1)	012346			
(2)	012366	000137	012320	
922	012372	000207		
923				
924				
925				
926	012374	012702	020036	
927	012400	012722	000000	
928	012404	022702	020130	
929	012410	001373		
930	012412	000207		
931				
932				
933				
934	012414	012702	020036	
935	012420	012722	004400	
936	012424	022702	020106	
937	012430	001373		
938	012432	000207		
939				
940				
941				
942	012434	016700	006426	
943	012440	004767	001512	
944	012444	012701	000120	
945	012450	132777	000200	006440
946	012456	001774		
947	012460	011077	006424	
948	012464	062700	000002	
949	012470	005301		
950	012472	001366		
951	012474	132777	000200	006414
952	012502	001401		
953	012504	000000		
954	012506	000207		
955				
956				
957				
958	012510	016767	006324	006360
959	012516	012700	000001	
960	012522	004767	001430	
961	012526	132777	000200	006362
962	012534	001774		
963	012536	012777	004400	006344
964	012544	005200		
965	012546	005367	006324	
966	012552	001365		
967	012554	132777	000200	006334
968	012562	001774		
969	012564	112777	000000	006320
970	012572	005200		
971	012574	022700	000121	
972	012600	001365		

```

CMP      GOOD,BAD
BEQ      .+4
ERR+GB   ;INTERRUPT REG - VALUE READ NOT EQUAL TO
LOOP     LP
JMP      @#LP
RTS      PC ;EXIT

;PUT SPACE CHAR IN ALL LOCS BF20 - BF1+10
CLRBFS: MOV      #BF20,R2
CB1:     MOV      #0,(R2)+
        CMP      #BF1+24,R2
        BNE     CB1
        RTS     PC

;PUT 1 CHAR (4400) IN LOCS BF20 - BF1
SETBFS: MOV      #BF20,R2
SB1:     MOV      #4400,(R2)+
        CMP      #BF1+2,R2
        BNE     SB1
        RTS     PC

;SEND 80 CHARS BEGINNING AT LOC POINTED TO BY PTR
P2SEND: MOV      PTR,R0
        JSR     PC,CKSWR
P2SD1:  MOV      #120,R1
        BITB   #200,@TIRHI
        BEQ    P2SD1 ;WAIT FOR OUTPUT DATA REQUEST
        MOV    (R0),@TDR ;SEND A CHAR
        ADD   #2,R0
        DEC   R1 ;ALL 80 CHARS SENT ?
        BNE  P2SD1 ;NO - GO SEND ANOTHER ONE
        BITB  #200,@TIRHI
        BEQ   .+4
        XX
        RTS   PC ;ERROR - GOT MORE THAN 80 ODR'S

;SEND 1 CHAR TO CARD #1 UP TO 20 CHARS TO CARD #20
P3SEND: MOV      CARD,TOG1
        MOV     #1,R0
        JSR     PC,CKSWR
P3SD1:  BITB   #200,@TIRHI
        BEQ    P3SD1 ;WAIT FOR OUTPUT DATA REQUEST
        MOV    #4400,@TDR ;SEND '4400' CHARACTER
        INC   R0
        DEC   TOG1
        BNE  P3SD1 ;NO - GO SEND ANOTHER ONE
P3SD2:  BITB   #200,@TIRHI
        BEQ    P3SD2 ;WAIT FOR OUTPUT DATA REQUEST
        MOVB  #0,@TDRHI ;SEND SPACE CHAR
        INC   R0
        CMP   #121,R0 ;HAVE ALL 80 CHARS BEEN SENT
        BNE  P3SD2 ;NO - GO SEND MORE
  
```



1005	012722	062767	000002	006132
1006	012730	005267	006142	
1007	012734	022767	000007	006134
1008	012742	001360		
1009	012744	132777	000200	006144
1010	012752	001774		
1011	012754	012777	000000	006126
1012	012762	005267	006110	
1013	012766	022767	000120	006102
1014	012774	001363		
1015	012776	000207		
1016				
1017				
1018				
1019		000002		
1020				
1021	013000	012700	000001	
1022	013004	016767	006042	006054
1023	013012	032777	000200	006074
1024	013020	001774		
1025	013022	032777	000002	006054
1026	013030	001402		
1027	013032	005267	006036	
1028	013036	017767	006024	006002
1029	013044	017767	006040	005764
1030	013052	026767	005770	005756
1031	013060	001401		
1032	013062	104007		
1033	013064	005767	006004	
1034	013070	001406		
1035	013072	022767	000377	005736
1036	013100	001407		
1037	013102	104007		
1038	013104	000405		
1039	013106	022767	000377	005722
1040	013114	001001		
1041	013116	104007		
1042				
1043	013120	005067	005750	
1044	013124	005200		
1045	013126	062767	000002	005732
1046	013134	022700	000121	
1047	013140	001324		
1048	013142	000207		
1049				
1050				
1051				
1052	013144	010067	000110	
1053	013150	005067	000100	
1054	013154	005067	000076	
1055	013160	005267	000070	
1056	013164	022767	000012	000062
1057	013172	001404		
1058	013174	005367	000060	
1059	013200	001367		
1060	013202	000407		

```

ADD #2,OUTPTR
INC TOG1
CMP #7,TOG1 ;HAVE ALL PRINTING CHARS BEEN SENT
BNE P4SD1 ;NO - GO PRINT MORE
P4SD2: BITB #200,@TIRHI
BEQ P4SD2 ;WAIT FOR OUTPUT DATA REQUEST
MOV #0,@TDR ;SEND SPACE CHAR
INC TOG1
CMP #120,TOG1 ;HAVE ALL 80 CHARS BEEN SENT
BNE P4SD2 ;NO - GO SEND MORE CHARS
RTS PC

;COMPARE INPUT DATA PATTERN BEGINNING AT INPTR
;
MP=2
CKDATA: MOV #1,R0
MOV INPTR,PTR
CD1: BIT #200,@TIR
BEQ CD1 ;WAIT FOR INPUT DATA REQUEST
BIT #MP,@TSR ;TEST FOR MULTI-PUNCH
BEQ 1$
INC MPSAV ;YES MP SET
1$: MOV @PTR,GOOD
MOV @TDR,BAD ;READ DATA REG
CMP GOOD,BAD
BEQ 2$
ERR+GB+COL ;DATA REG - COMPARE ERROR AT INDICATED C
2$: TST MPSAV ;LOOK FOR MULTI-P BIT
BEQ 3$ ;NO
CMP #377,BAD ;IS DATA MULTI-DATA
BEQ 5$ ;YES
ERR+GB+COL ;MULTI-P SET,DATA NOT 377
BR 5$ ;EXIT
3$: CMP #377,BAD ;TEST FOR MULTI-DATA
BNE 5$ ;NO MULTI-DATA
ERR+GB+COL ;MULTI-P DATA,NO MP SEEN
5$: CLR MPSAV
INC R0
ADD #2,PTR
CMP #121,R0
BNE CD1 ;IF MORE COLUMNS TO READ GO DO IT
RTS PC ;EXIT

;CONVERT OCTAL # IN R0 TO DECIMAL # AND TYPE OUT DECIMAL #
;
OCTDEC: MOV R0,TOG
CLR ONES
CLR TENS
OD1: INC ONES
CMP #12,ONES
BEQ OD2
DEC TOG
BNE OD1
BR OD3
  
```

1061	013204	005067	000044
1062	013210	005267	000042
1063	013214	005367	000040
1064	013220	001357	
1065	013222	016704	000030
1066	013226	062704	000060
1067	013232	004767	001556
1068	013236	016704	000012
1069	013242	062704	000060
1070	013246	004767	001542
1071	013252	000207	
1072	013254	000000	
1073	013256	000000	
1074	013260	000000	
1075			
1076	013262	104400	
1077	013264	015116	
1078	013266	104400	
1079	013270	015164	
1080	013272	104406	
1081	013274	021100	
1082	013276	104400	
1083	013300	015213	
1084	013302	104403	
1085	013304	021120	
1086	013306	104400	
1087	013310	015257	
1088	013312	104401	
1089	013314	021124	
1090			
1091	013316	104400	
1092	013320	015311	
1093	013322	004767	001542
1094	013326	022704	000131
1095	013332	001004	
1096	013334	004767	001454
1097	013340	000167	000014
1098	013344	022704	000116
1099	013350	001364	
1100	013352	004767	001436
1101	013356	000207	
1102			
1103	013360	104400	
1104	013362	015377	
1105	013364	104400	
1106	013366	015164	
1107	013370	012705	000006
1108	013374	004767	000470
1109	013400	042704	000007
1110	013404	012705	021100
1111	013410	042715	177770
1112	013414	050425	
1113	013416	020527	021120
1114	013422	001372	
1115	013424	104400	
1116	013426	015213	

```
OD2: CLR ONES
      INC TENS
      DEC TOG
      BNE OD1
OD3: MOV TENS,R4
      ADD #60,R4
      JSR PC,TTYOUT ;TYPEOUT TENS
      MOV ONES,R4
      ADD #60,R4
      JSR PC,TTYOUT ;TYPEOUT ONES
      RTS PC
ONES: XX
TENS: XX
TOG: XX
;INPUT ROUTINE FOR DEVICE ADDRESS, INT VECTOR ADDRESS AND BR LEV
DEVCOD: TYPE
        DCMSG1 ;"THE FOLLOWING VALUES ARE BEING USED"
        TYPE
        DCMSG2 ;" DEVICE ADDRESS = "
        TYPE+6 ;TYPE TCR ADDRESS
        TCR ;TYPE TCR ADDRESS
        TYPE
        DCMSG3 ;" PC INTERRUPT VECTOR ADDRESS = "
        TYPE+3 ;TYPE PCV ADDRESS
        PCV ;TYPE PCV ADDRESS
        TYPE
        DCMSG4 ;" BUS REQUEST LEVEL = "
        TYPE+1 ;TYPE BRLV
        BRLV ;TYPE BRLV
;TEST TO SEE IF VALUES ARE TO BE CHANGED
        TYPE
        DCMSG5 ;"DO YOU WANT TO CHANGE ANY OF THESE VAL
DC1: JSR PC,TTYIN ;INPUT A CHAR
      CMP #131,R4 ;IS IT Y ?
      BNE DC2 ;NO - GO CHECK FOR N
      JSR PC,TTYOUT ;YES - ECHO Y
      JMP DC3 ;GO GET NEW VALUES
DC2: CMP #116,R4 ;IS IT N ?
      BNE DC1 ;NO - GO BACK TO INPUT
      JSR PC,TTYOUT ;YES - ECHO N
      RTS PC ;EXIT
;VALUES ARE TO BE CHANGED
DC3: TYPE
      DCMSG6 ;" ENTER VALUE AFTER (=) IS TYPED"
      TYPE
      DCMSG2 ;" DEVICE ADDRESS = "
      MOV #6,R5 ;GET NEW DEVICE ADDRESS
      JSR PC,RDOCT
      BIC #7,R4
      MOV #TCR,R5
DC4: BIC #177770,(R5) ;STORE 4 NEW DEVICE ADDRESSES
      BIS R4,(R5)+
      CMP R5,#PCV
      BNE DC4
      TYPE
      DCMSG3 ;" PC INTERRUPT VECTOR ADDRESS = "
```

1117 013430 012705 000003  
1118 013434 004767 000430  
1119 013440 010467 005454  
1120 013444 062704 000002  
1121 013450 010467 005446  
1122 013454 104400  
1123 013456 015257  
1124 013460 012705 000001  
1125 013464 004767 000400  
1126 013470 010467 005430  
1127 013474 000207  
1128  
1129  
1130  
1131 013476 004767 000454  
1132 013502 032777 020000 165170  
1133 013510 001403  
1134 013512 012767 000001 005350  
1135 013520 011667 005330  
1136 013524 032777 040000 165146  
1137 013532 001041  
1138 013534 104400  
1139 013536 015576  
1140 013540 011667 005270  
1141 013544 162767 000002 005262  
1142 013552 104406  
1143 013554 021034  
1144 013556 032777 000002 005250  
1145 013564 001404  
1146 013566 104400  
1147 013570 015622  
1148 013572 104406  
1149 013574 021046  
1150 013576 032777 000001 005230  
1151 013604 001404  
1152 013606 104400  
1153 013610 015631  
1154 013612 104406  
1155 013614 021036  
1156 013616 032777 000004 005210  
1157 013624 001404  
1158 013626 104400  
1159 013630 015640  
1160 013632 004767 177306  
1161 013636 012704 000015  
1162 013642 004767 001146  
1163 013646 032777 100000 165024  
1164 013654 001001  
1165 013656 000000  
1166 013660 000002  
1167  
1168  
1169  
1170 013662 011603  
1171 013664 162703 000002  
1172 013670 032713 000007

```
MOV #3,R5  
JSR PC,RDOCT ;GET NEW INT VECTOR ADDRESS  
MOV R4,PCV ;SET NEW PC VECTOR ADDRESS  
ADD #2,R4  
MOV R4,PSV ;SET NEW PS VECTOR ADDRESS  
TYPE  
DCMSG4 ;" BUS REQUEST LEVEL = "  
MOV #1,R5  
JSR PC,RDOCT ;GET NEW BR LEVEL  
MOV R4,BRLV ;SET NEW BR LEVEL  
RTS PC ;EXIT  
:  
:ERROR SUBROUTINE  
:  
ERROR: JSR PC,CKSWR  
BIT #20000,@SWR  
BEQ .+10 ;IS LOOP ON ERROR SELECTED ?  
MOV #1,SCOPE ;YES - SET SCOPE TO A 1  
MOV (SP),LASTPC ;PUT ADDRESS OF ERROR IN LASTPC  
BIT #40000,@SWR ;IF SR14=1 DELETE TYPEOUT  
BNE E3 ;EXIT NOW  
TYPE  
PCMSG ;"ERROR AT ADDRESS"  
MOV (SP),ADR  
SUB #2,ADR  
TYPE+6  
ADR ;TYPE CONTENTS OF ADR  
BIT #2,@ADR ;TYPE GOOD ?  
BEQ E1 ;NO  
TYPE  
GDMSG ;"GOOD="  
TYPE+0  
GOOD ;TYPE CONTENTS OF GOOD  
E1: BIT #1,@ADR ;TYPE BAD ?  
BEQ E2 ;NO  
TYPE  
BDMSG ;"BAD ="  
TYPE+6  
BAD ;TYPE CONTENTS OF BAD  
E2: BIT #4,@ADR  
BEQ E3  
TYPE  
COLMSG ;"COLUMN #="  
JSR PC,OCTDEC ;TYPEOUT DECIMAL COLUMN # IN ERROR  
E3: MOV #15,R4  
JSR PC,TTYOUT ;TYPE CR + LF  
BIT #100000,@SWR ;IF SR15=1  
BNE E4 ;BYPASS THE ERROR HALT  
ERRHLT: XX ;ERROR HALT  
E4: RTS ;EXIT  
:  
:DETERMINE IF THIS TYPEOUT IS FOR MESSAGE OR OCTAL VALUE AND BRANCH ACCO  
:  
TYPOUT: MOV (SP),R3 ;GET PC FROM STACK  
SUB #2,R3 ;SUBTRACT 2  
BIT #7,(R3) ;TEST LOW ORDER 3 BITS OF TRAP INSTRUCTI
```

```
1173 013674 001401          BEQ      TYPMSG      ;IF 0'S GO TO TYPMSG
1174 013676 000421          BR       TYPOCT      ;IF NOT 0'S GO TO TYPOCT
1175
1176          ;MESSAGE TYPEOUT - R5=ADDRESS OF BEGINNING OF MESSAGE
1177 013700 004767 000252  TYPMSG: JSR      PC,CKSWR
1178 013704 011604          MOV      (SP),R4
1179 013706 011405          MOV      (R4),R5      ;R5=ADDRESS OF MESSAGE
1180 013710 112504          TM1:    MOVB     (R5)+,R4  ;PUT CHAR IN R4
1181 013712 001410          BEQ      TM3         ;IF IT EQUALS 0 TERMINATE
1182 013714 022704 000046  CMP      #46,R4      ;DOES CHAR = CRLF FLAG ?
1183 013720 001002          BNE     TM2         ;NO - GO TYPE CHAR
1184 013722 012704 000015  MOV      #15,R4      ;YES - TYPE CR+LF
1185 013726 004767 001062  TM2:    JSR      PC,TTYOUT ;GO TYPE CONTENTS OF R4
1186 013732 000766          BR       TM1         ;GO GET NEXT CHAR
1187 013734 062716 000002  TM3:    ADD      #2,(SP) ;ADD 2 TO SP FOR RETURN
1188 013740 000002          RTI                ;EXIT
1189
1190          ;OCTAL TYPEOUT - R4=VALUE R5=# OF DIGITS TO TYPE
1191
1192 013742 011305          TYPOCT: MOV      (R3),R5
1193 013744 042705 177770  BIC      #177770,R5   ;R5 = # OF DIGITS TO BE TYPED
1194 013750 062703 000002  ADD      #2,R3
1195 013754 011367 005104  MOV      (R3),PT
1196 013760 017704 005100  MOV      @PT,R4      ;R4 = VALUE TO BE TYPED
1197 013764 010567 005052  MOV      R5,CNT      ;SET CNT FOR STACKING
1198 013770 010467 005076  TO.1:   MOV      R4,TEMP     ;PUT VALUE IN TEMP
1199 013774 042767 177770 005070  BIC      #177770,TEMP ;MASK OUT ALL BUT BITS 2,1+0
1200 014002 062767 000060 005062  ADD      #60,TEMP    ;MAKE ASCII NUMBER
1201 014010 016746 005056  MOV      TEMP,-(SP)  ;STORE NUMBER ON STACK
1202 014014 006004          ROR      R4
1203 014016 006004          ROR      R4
1204 014020 006004          ROR      R4
1205 014022 005367 005014  DEC      CNT         ;IF CNT IS NOT ZERO
1206 014026 001360          BNE     TO.1        ;GO STACK ANOTHER NUMBER
1207 014030 022705 000006  CMP      #6,R5
1208 014034 001002          BNE     TO.2        ;IF 6 NUMBERS WERE SELECTED
1209 014036 042716 000006  BIC      #6,(SP)     ;CLEAR BITS 1+0 OF LAST NUMBER STACKED
1210 014042 010567 004774  TO.2:   MOV      R5,CNT      ;SET CNT FOR TYPING
1211 014046 012604          TO.3:   MOV      (SP)+,R4  ;GET A NUMBER FROM THE STACK
1212 014050 004767 000740  JSR      PC,TTYOUT  ;TYPE THE NUMBER
1213 014054 005367 004762  DEC      CNT         ;IF COUNT IS NOT ZERO
1214 014060 001372          BNE     TO.3        ;GO TYPE ANOTHER NUMBER
1215 014062 062716 000002  ADD      #2,(SP)    ;ADD 2 TO SP FOR RETURN
1216 014066 000002          RTI                ;EXIT
1217 014070          S7$::
1218          ;OCTAL TYPEIN - R5=#OF DIGITS TO BE INPUT INTO R4
1219
1220 014070 010567 004746  RDOCT:  MOV      R5,CNT      ;CNT=# OF DIGITS TO BE INPUT
1221 014074 005067 004772  CLR      TEMP
1222 014100 004767 000764  RD.1:  JSR      PC,TTYIN   ;GET CHAR FROM TTY
1223 014104 004767 000704  JSR      PC,TTYOUT  ;ECHO IT
1224 014110 042704 177770  BIC      #177770,R4  ;MAKE CHAR OCTAL
1225 014114 006167 004752  ROL      TEMP
1226 014120 006167 004746  ROL      TEMP
1227 014124 006167 004742  ROL      TEMP
1228 014130 042767 000007 004734  BIC      #7,TEMP    ;CLEAR TEMP BITS 2,1+0
```

1229	014136	060467	004730	
1230	014142	005367	004674	
1231	014146	001354		
1232	014150	016704	004716	
1233	014154	000207		
1234				
1235				
1236				
1237				
1238				
1239	014156	022767	000176	164514
1240	014164	001075		
1241	014166	105767	163366	
1242	014172	100072		
1243	014174	116746	163362	
1244	014200	042716	177600	
1245	014204	022726	000007	
1246	014210	001063		
1247	014212	126727	164474	000001
1248	014220	001457		
1249	014222	104400	014463	
1250	014226	104400	014470	
1251	014232	016746	163740	
1252	014236	004767	000300	
1253	014242	104400	014500	
1254	014246	005046		
1255	014250	005046		
1256	014252	105767	163302	
1257	014256	100375		
1258	014260	116746	163276	
1259	014264	042716	177600	
1260	014270	021627	000025	
1261	014274	001005		
1262	014276	104400	014456	
1263	014302	062706	000006	
1264	014306	000757		
1265	014310	021627	000015	
1266	014314	001022		
1267	014316	005766	000004	
1268	014322	001403		
1269	014324	016677	000002	164346
1270	014332	062706	000010	
1271				
1272				
1273	014336	104400	014512	
1274	014342	126727	164345	000001
1275	014350	001003		
1276	014352	012767	000100	163200
1277	014360	000207		
1278	014362	004767	000356	
1279	014366	021627	000060	
1280	014372	002420		
1281	014374	021627	000067	
1282	014400	003015		
1283	014402	042726	000060	
1284	014406	005766	000002	

```

ADD R4,TEMP ;ADD INPUT CHAR TO OTHERS
DEC CNT ;IF ALL CHARS NOT READ IN
BNE RD.1 ;GO READ ANOTHER ONE
MOV TEMP,R4 ;PUT FINAL OCTAL VALUE IN R4
RTS PC ;EXIT

;
; *SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
; *ROUTINE IS ENTERED FROM TRAP HANDLERR, AND WILL
; *SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
; *WHEN OPERATING IN TTY FLAG MODE.
CKSWR: CMP #SWREG,SWR ;IS THE SOFT-SWR SELECTED?
      BNE K15 ;BRANCH IF NO
      TSTB TKS ;CHAR THERE?
      BPL K15 ;IF NO,DON'T WAIT AROUND
      MOVB TKB,-(SP) ;SAVE THE CHAR
      BIC #^C177,(SP) ;STRIP OFF THE ASCII
      CMP #7,(SP)+ ;IS IT A CONTROL G?
      BNE K15 ;NO RETURN TO USER
      CMPB AUTOB,#1 ;ARE WE RUNNING IN AUTO-MODE?
      BEQ K15 ;BRANCH IF YES
      TYPE ,CNTLG ;ECHO THE CONTROL-G (^G)
GTSWR: TYPE ,MSWR ;TYPE CURRENT CONTENTS
      MOV SWREG,-(SP) ;SAVE SWREG FOR TYPEOUT
      JSR PC,TYPEOC ;GO TYPE-OCTAL ASCII(ALL DIGITS)
      TYPE ,MNEW ;PROMPT FOR NEW SWR
K19: CLR -(SP) ;CLEAR COUNTER
      CLR -(SP) ;NEW SWR
K7: TSTB TKS ;CHAR THERE?
      BPL K7 ;IF NOT TRY AGAIN
      MOVB TKB,-(SP) ;PICK UP CHAR
      BIC #^C177,(SP) ;MAKE IT 7-BIT ASCII
K9: CMP (SP),#25 ;IT IS A CONTROL-U?
      BNE K10 ;BRANCH IF NOT
      TYPE ,CNTLU ;YES ECHO CONTROL - U(^U)
K20: ADD #6,SP ;IGNORE PREVIOUS INPUT
      BR K19 ;LET'S TRY IT AGAIN
K10: CMP (SP),#15 ;IS IT A <CR>?
      BNE K16 ;BRANCH IF NO
      TST 4(SP) ;YES,IS IT THE FIRST CHAR?
      BEQ K11 ;BRANCH IF YES
      MOV 2(SP),@SWR ;SAVE NEW SWR
K11: ADD #10,SP ;CLEAR UP STACK
; TTY INPUT ROUTINE
;
K14: TYPE ,CRLF ;ECHO <CR> AND <LF>
      CMPB INTAG,#1 ;RE-ENABLE TTY KBD INTERRUPTS
      BNE K15 ;BRANCH IF NOT
      MOV #100,TKS ;RE-ENABLE TTY KBD INTERRUPTS
K15: RTS PC ;RETURN
K16: JSR PC,TYPEC ;ECHO CHAR
      CMP (SP),#60 ;CHAR < 0?
      BLT K18 ;BRANCH IF YES
      CMP (SP),#67 ;CHAR > 7?
      BGT K18 ;BRANCH IF YES
      BIC #60,(SP)+ ;STRIP-OFF ASCII
      TST 2(SP) ;IS THIS THE FIRST CHAR

```



1285 014412 001403  
1286 014414 006316  
1287 014416 006316  
1288 014420 006316  
1289 014422 005266 000002  
1290 014426 056616 177776  
1291 014432 000707  
1292 014434 104400 014511  
1293 014440 000720  
1294 014442 000014  
1295  
1296 014456 052536 005015 000  
1297 014463 136 006507 000012  
1298 014470 005015 053523 036522  
1299 014476 000040  
1300 014500 020040 042516 020127  
1301 014506 020075 000  
1302 014511 077  
1303 014512 015  
1304 014513 012 000  
1305  
1306  
1307  
1308  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1330  
1331 014516 014516  
1332 014516 017646 000000  
1333 014522 116667 000001 000211  
1334 014530 112667 000207  
1335 014534 062716 000002  
1336 014540 000406  
1337 014542 112767 000001 000171  
1338 014550 112767 000006 000165

```
BEQ K17 ;BRANCH IF YES
ASL (SP) ;NO SHIFT PRESENT
ASL (SP) ;CHAR OVER TO MAKE
ASL (SP) ;ROOM FOR NEW ONE
K17: INC 2(SP) ;KEEP COUNT OF CHARACTER
BIS -2(SP),(SP) ;SET IN NEW CHAR
BR K7 ;GET THE NEXT ONE
K18: TYPE ,QUES ;TYPE ?<CR><LF>
BR K20 ;SIMULATE CONTROL-U
TTYN: .BLKB 12. ;RESERVE 12 BYTE FOR TTYT
;Y INPUT
CNTLU: .ASCIZ /^U/<15><12> ;CONTROL 'U'
CNTLG: .ASCIZ /^G/<15><12> ;CONTROL 'G'
MSWR: .ASCIZ <15><12>/SWP= /

MNEW: .ASCIZ / NEW = /

QUES: .ASCII /?/ ;QUESTION MARK
CRLF: .ASCII <15> ;CARRIAGE RETURN
LF: .ASCIZ <12> ;LINE FEED
;*****
;THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO
;A 6-DIGIT OCTAL (ASCII) NUMBER AND TYPE IT
;*TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF
;*DIGITS TO TYPE
;*CALL:
;* MOV NUM,-(SP) ;NUMBER TO BE TYPED
;* TYPOS ;CALL FOR TYPE OUT
;* .BYTE N ;N=1 TO 6 FOR NUMBER OF DIGITS
;* ;TO TYPE
;* .BYTE M ;M=1 OR 0
; ;1=TYPE LEADING ZEROS
; ;0=SUPPRESS LEADING ZEROS
;*
;*TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS
;*THE LAST
;*TYPOS OR TYPOC
;*CALL:
;* MOV NUM,-(SP) ;NUMBER TO BE TYPED
;* TYPON ;CALL FOR TYPEOUT
;*
;*TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
;*CALL:
;* MOV NUM,-(SP) ;NUMBER TO BE TYPED
;* TYPOC ;CALL FOR TYPEOUT

.EVEN
TYPOS: MOV @ (SP),-(SP) ;PICKUP THE MODE
MOV B 1(SP),FILL0 ;LOAD ZERO FILL SWITCH
MOV B (SP)+,MODE0+1 ;NUMBER OF DIGIT TO TYPE
ADD #2,(SP) ;ADJUST RETURN ADDRESS
BR TYPON
TYPOC: MOV B #1,FILL0 ;SET THE ZERO FILL SWITCH
MOV B #6,MODE0+1 ;SET FOR SIX(6) DIGITS
```

.MA  
CZC  
ADR  
AEN  
APA  
ASW  
AUT  
A1  
A2  
A3  
A4  
A5  
A6  
A7  
A8  
B  
BAD  
BC1  
BDM  
BF1  
BF10  
BF11  
BF12  
BF13  
BF14  
BF15  
BF16  
BF17  
BF18  
BF19  
BF2  
BF20  
BF3  
BF4  
BF5  
BF6  
BF7  
BF8  
BF9  
BLK  
BRL  
B1  
CAR  
CAR  
CAR  
CAR  
CAR  
CAR  
CAR  
CAR



1395	014774	000406		
1396	014776	122766	014513	000002
1397	015004	001402		
1398	015006	105227		
1399	015010	000000		
1400	015012	000207		
1401				
1402				
1403				
1404				
1405				
1406	015014	032777	040000	163656
1407	015022	001021		
1408	015024	010467	162536	
1409	015030	105767	162530	
1410	015034	100375		
1411	015036	022704	000015	
1412	015042	001003		
1413	015044	012704	000012	
1414	015050	000761		
1415	015052	012767	040000	004000
1416	015060	005367	003774	
1417	015064	001375		
1418	015066	000207		
1419				
1420				
1421				
1422	015070	005267	162464	
1423	015074	032767	000200	162456
1424	015102	001774		
1425	015104	016704	162452	
1426	015110	042704	177600	
1427	015114	000207		
1428	015116	023046	044124	020105
	015124	047506	046114	053517
	015132	047111	020107	040526
	015140	052514	051505	040440
	015146	042522	041040	044505
	015154	043516	052440	042523
	015162	000104		
1429	015164	020046	020040	042040
	015172	053105	041511	020105
	015200	042101	051104	051505
	015206	020123	020075	000
1430	015213	046	020040	020040
	015220	041520	044440	052116
	015226	051105	052522	052120
	015234	053040	041505	047524
	015242	020122	042101	051104
	015250	051505	020123	020075
	015256	060		
1431	015257	046	020040	020040
	015264	052502	020123	042522
	015272	052521	051505	020124
	015300	042514	042526	020114
	015306	020075	000	

```

B1: BR TYPEX ;EXIT
    CMPB #LF,2(SP) ;IS CHAR LINE FEED?
    BEQ TYPEX ;BRANCH IF YES
    INCB (PC)+ ;COUNT THE CHARS
CHARCNT: .WORD 0 ;CHARACTER COUNT SPACE
TYPEX: RTS PC

;TYPE THE CHARACTER IN R4
TTYOUT: BIT #40000,@SWR ;IF SR14=1 DELETE TYPEOUT
        BNE TTY.3
        MOV R4,TPB ;TYPE CHAR IN R4
TTY.1: TSTB TPS
        BPL TTY.1 ;WAIT FOR DONE
        CMP #15,R4 ;WAS IT A CR ?
        BNE TTY.2 ;IF NO - GO EXIT
        MOV #12,R4 ;IF YES - PUT LF IN R4
        BR TTYOUT ;GO TYPE THE LF
TTY.2: MOV #40000,LOC
        DEC LOC
        BNE TTY.3
TTY.3: RTS PC ;EXIT

;READ CHAR FROM TTY INTO R4
TTYIN: INC TKS ;FETCH A CHAR
TTY.1: BIT #200,TKS
        BEQ TTY.1 ;WAIT FOR DONE FLAG
        MOV TKB,R4 ;READ CHAR INTO R4
        BIC #177600,R4 ;MASK R4 WITH ALL BUT 177
        RTS PC ;EXIT
DCMSG1: .ASCIZ /&&THE FOLLOWING VALUES ARE BEING USED/

DCMSG2: .ASCIZ /& DEVICE ADDRESS = /

DCMSG3: .ASCIZ /& PC INTERRUPT VECTOR ADDRESS = /

DCMSG4: .ASCIZ /& BUS REQUEST LEVEL = /

```

.MA  
 CZC  
 ERR  
 E1  
 E2  
 E3  
 E4  
 FIL  
 G  
 GB  
 GDM  
 GOO  
 GTS  
 HOL  
 INI  
 INI  
 INI  
 INP  
 INT  
 K10  
 K11  
 K14  
 K15  
 K16  
 K17  
 K18  
 K19  
 K20  
 K7  
 K9  
 LAS  
 LEV  
 LF  
 LOC  
 LOG  
 LP

1432 015311 046 042046 020117  
015316 047531 020125 040527  
015324 052116 052040 020117  
015332 044103 047101 042507  
015340 040440 054516 047440  
015346 020106 044124 051505  
015354 020105 040526 052514  
015362 051505 024040 020131  
015370 051117 047040 037451  
015376 000  
1433 015377 046 020040 047105  
015404 042524 020122 040526  
015412 052514 020105 043101  
015420 042524 020122 036450  
015426 020051 051511 052040  
015434 050131 042105 000  
1434 015441 046 025046 025052  
015446 025052 046040 043517  
015454 041511 052040 051505  
015462 051524 025040 025052  
015470 025052 023046 000  
1435 015475 046 050046 052125  
015502 034040 031460 026465  
015510 030070 032464 047440  
015516 043106 046055 047111  
015524 026105 051040 046505  
015532 053117 020105 046101  
015540 020114 040503 042122  
015546 000123  
1436 015550 050046 042522 051523  
015556 041440 020122 047524  
015564 041440 047117 044524  
015572 052516 000105  
1437 015576 023046 051105 047522  
015604 020122 052101 040440  
015612 042104 042522 051523  
015620 000040  
1438 015622 043446 047517 036504  
015630 000  
1439 015631 046 040502 020104  
015636 000075  
1440 015640 041446 046117 046525  
015646 020116 036443 000  
1441 015653 046 050046 052125  
015660 034040 031460 026465  
015666 030070 032464 047440  
015674 026516 044514 042516  
015702 040440 042116 050040  
015710 042522 051523 051440  
015716 040524 052122 047440  
015724 020116 044124 020105  
015732 030070 032463 034055  
015740 032060 000065  
1442 015744 023046 052520 020124  
015752 030070 032463 034055  
015760 032060 020065 043117

DCMSG5: .ASCIZ /&&DO YOU WANT TO CHANGE ANY OF THESE VALUES (Y OR N)?/

DCMSG6: .ASCIZ /& ENTER VALUE AFTER (=) IS TYPED/

LOGTST: .ASCIZ /&&\*\*\*\*\* LOGIC TESTS \*\*\*\*\*&&/

INITM1: .ASCIZ /&&PUT 8035-8045 OFF-LINE, REMOVE ALL CARDS/

CONMSG: .ASCIZ /&PRESS CR TO CONTINUE/

PCMSG: .ASCIZ /&&ERROR AT ADDRESS /

GDMSG: .ASCIZ /&GOOD-/

BDMSG: .ASCIZ /&BAD =/

COLMSG: .ASCIZ /&COLUMN #-/

T2MSG1: .ASCIZ /&&PUT 8035-8045 ON-LINE AND PRESS START ON THE 8035-804

T5M1: .ASCIZ /&&PUT 8035-8045 OFF-LINE AND PRESS START ON THE 8035\_80

MNE  
MOD  
MP  
MPS  
MSW  
OCT  
OD1  
OD2  
OD3  
ONE  
OUT  
PAS  
PAS  
PAS  
PCM  
PCV  
PRO  
PS  
PSV  
PT  
PTR  
P1A  
P1B  
P1D  
P1E  
P1I  
P1M  
P1M  
P1M  
P2A  
P2B  
P2C  
P2I  
P2M  
P2M  
P2M  
P2S  
P2S  
P3A  
P3M  
P3M  
P3S  
P3S  
P3S  
P4A  
P4M  
P4M  
P4M  
P4S  
P4S  
P5S  
P5S

	015766	026506	044514	042516
	015774	040440	042116	050040
	016002	042522	051523	051440
	016010	040524	052122	047440
	016016	020116	044124	020105
	016024	030070	032463	034137
1443	016032	032060	000065	
	016036	023046	051120	051505
	016044	020123	052123	051101
	016052	020124	047117	052040
	016060	042510	034040	031460
	016066	026465	030070	032464
	016074	000		
1444	016075	046	050046	046125
	016107	020114	040503	042122
	016111	050040	040514	042524
	016116	047440	020106	052123
	016124	041501	042513	020122
	016132	030443	041040	041501
	016140	020113	047524	050040
	016146	051517	052111	047511
	016154	020116	044127	041511
	016162	000110		
1445	016164	053446	046111	020114
	016172	044523	052515	040514
	016200	042524	051440	040524
	016206	045503	051105	043040
	016214	046125	020114	047101
	016222	020104	042522	042514
	016230	051501	000105	
1446	016234	050046	047522	051107
	016242	046501	053440	046111
	016250	020114	040527	052111
	016256	043040	051117	051440
	016264	040524	045503	051105
	016272	043040	046125	020114
	016300	046106	043501	000
1447	016305	046	050046	042522
	016312	051523	051440	040524
	016320	052122	047440	020116
	016326	044124	020105	030070
	016334	032463	034055	032060
	016342	000065		
1448	016344	023046	052520	046114
	016352	041440	051101	020104
	016360	046120	052101	020105
	016366	043117	051440	040524
	016374	045503	051105	021440
	016402	020062	040502	045503
	016410	052040	020117	047520
	016416	044523	044524	047117
	016424	053440	044510	044103
	016432	000		
1449	016433	046	050046	042522
	016440	051523	051440	040524
	016446	052122	047440	020116

T5MSG: .ASCIZ /BBPRESS START ON THE 8035-8045/

T5MSG1: .ASCIZ /BBPULL CARD PLATE OF STACKER #1 BACK TO POSITION WHICH/

T5MSG2: .ASCIZ /BWILL SIMULATE STACKER FULL AND RELEASE/

T5MSG3: .ASCIZ /BPROGRAM WILL WAIT FOR STACKER FULL FLAG/

T5MSG4: .ASCIZ /BBPRESS START ON THE 8035-8045/

T5MSG5: .ASCIZ /BBPULL CARD PLATE OF STACKER #2 BACK TO POSITION WHICH/

T5MSG6: .ASCIZ /BBPRESS START ON THE 8035-8045/

.MA  
CZC  
PSS  
QUE  
RDO  
RD.  
RT1  
RT2  
RT3  
RT4  
SB1  
SCO  
SET  
SR  
STA  
ST1  
ST2  
ST3  
SWR  
SWR  
SWR  
S7\$  
TCR  
TCR  
TDR  
TDR  
TEM  
TEN  
TES  
TES  
TES  
TES  
TIR  
TIR  
TKB  
TKS  
TM1  
TM2  
TM3  
TOG  
TOG



1457	017136	050046	040514	042503	P2MSG2: .ASCIZ /&PLACE ALL 20 CARDS IN THE PRIMARY HOPPER/
	017144	040440	046114	031040	
	017152	020060	040503	042122	
	017160	020123	047111	052040	
	017166	042510	050040	044522	
	017174	040515	054522	044040	
	017202	050117	042520	000122	
1458	017210	050046	042522	051523	P2MSG3: .ASCIZ /&PRESS STOP-RESET AND PRESS START ON THE 8035-8045/
	017216	051440	047524	026520	
	017224	042522	042523	020124	
	017232	040440	042116	050040	
	017240	042522	051523	051440	
	017246	040524	052122	047440	
	017254	020116	044124	020105	
	017262	030070	032463	034055	
	017270	032060	000065		
1459	017274	050046	042522	051523	P2MSG4: .ASCIZ /&PRESS STOP-RESET ON 8035-8045 /
	017302	051440	047524	026520	
	017310	042522	042523	020124	
	017316	047117	034040	031460	
	017324	026465	030070	032464	
	017332	000040			
1460	017334	053046	051105	043111	P3MSG1: .ASCIZ /&VERIFY THAT ALL CHARS WERE PUNCHED ON EACH CARD IN A/
	017342	020131	044124	052101	
	017350	040440	046114	041440	
	017356	040510	051522	053440	
	017364	051105	020105	052520	
	017372	041516	042510	020104	
	017400	047117	042440	041501	
	017406	020110	040503	042122	
	017414	044440	020116	000101	
1461	017422	050046	042522	042503	P3MSG2: .ASCIZ /&PRECESS FASHION BEGINNING W TH COLUMN 11 OF 1ST CARD/
	017430	051523	043040	051501	
	017436	044510	047117	041040	
	017444	043505	047111	044516	
	017452	043516	053440	052111	
	017460	020110	047503	052514	
	017466	047115	030440	020061	
	017474	043117	030440	052123	
	017502	041440	051101	000104	
1462	017510	053046	051105	043111	P4MSG1: .ASCIZ /&VERIFY THAT THERE IS A ALL A'S CHAR PUNCHED IN CARD 1/
	017516	020131	044124	052101	
	017524	052040	042510	042522	
	017532	044440	020123	020101	
	017540	046101	020114	023501	
	017546	020123	044103	051101	
	017554	050040	047125	044103	
	017562	042105	044440	020116	
	017570	040503	042122	030440	
	017576	000			
1463	017577	046	044124	052101	P4MSG2: .ASCIZ /&THAT IS HOLE PUNCHED IN ROW 12 AND 1/
	017604	044440	020123	047510	
	017612	042514	050040	047125	
	017620	044103	042105	044440	
	017626	020116	047522	020127	
	017634	031061	040440	042116	

1464 017642 030440 000  
017645 046 050125 052040  
017652 020117 030062 021040  
017660 046101 020114 023501  
017666 021123 041440 040510  
017674 051522 050040 047125  
017702 044103 042105 044440  
017710 020116 040503 042122  
017716 031040 000060  
1465 017722 023046 047105 020104  
017730 043117 052040 051505  
017736 044524 043516 023046  
017744 000  
1466 017745 046 020046 042504  
017752 051503 042520 026503  
017760 030461 041455 041532  
017766 041124 026501 052103  
017774 030523 026461 041512  
020002 044055 046117 042514  
020010 044522 044124 030440  
020016 026462 044502 051524  
020024 052055 051505 020124  
020032 023046 000  
1467  
1468 020036  
1469 020036 000000  
1470 020040 000000  
1471 020042 000000  
1472 020044 000000  
1473 020046 000000  
1474 020050 000000  
1475 020052 000000  
1476 020054 000000  
1477 020056 000000  
1478 020060 000000  
1479 020062 000000  
1480 020064 000000  
1481 020066 000000  
1482 020070 000000  
1483 020072 000000  
1484 020074 000000  
1485 020076 000000  
1486 020100 000000  
1487 020102 000000  
1488 020104 000000  
1489 020106 000000  
1490 020110 000000  
1491 020112 000000  
1492 020114 000000  
1493 020116 000000  
1494 020120 000000  
1495 020122 000000  
1496 020124 000000  
1497 020126 000000  
1498 020130 004006  
1499 020132 000006

P4MSG3: .ASCIZ /&UP TO 20 'ALL A'S' CHARS PUNCHED IN CARD 20/

ENDMSG: .ASCIZ /&&END OF TESTING&&/

PROGM: .ASCIZ /&& DECSPEC-11-CZCTBA-CTS11-JC-HOLLERITH 12-BITS-TEST &&/

.EVEN  
BF20: 0  
BF19: 0  
BF18: 0  
BF17: 0  
BF16: 0  
BF15: 0  
BF14: 0  
BF13: 0  
BF12: 0  
BF11: 0  
BF10: 0  
BF9: 0  
BF8: 0  
BF7: 0  
BF6: 0  
BF5: 0  
BF4: U  
BF3: 0  
BF2: 0  
BF1: 0

4006  
6

.MA  
CZC  
CKT  
CKT  
CKT  
CKT  
LOO  
WTD  
WTI  
WTO  
. A  
ER  
CZ  
RU  
RU  
CO



1500	020134	000102	102	:#
1501	020136	001042	1042	:X
1502	020140	004000	4000	:B
1503	020142	000022	22	:.
1504	020144	004022	4022	:(
1505	020146	002022	2022	:)
1506	020150	002042	2042	:*
1507	020152	004012	4012	:+
1508	020154	001102	1102	:.
1509	020156	002000	2000	:-
1510	020160	001400	1400	:/
1511	020162	001000	1000	:0
1512	020164	000400	400	:1
1513	020166	000200	200	:2
1514	020170	000100	100	:3
1515	020172	000040	40	:4
1516	020174	000020	20	:5
1517	020176	000010	10	:6
1518	020200	000004	4	:7
1519	020202	000002	2	:8
1520	020204	000001	1	:9
1521	020206	000202	202	:.
1522	020210	002012	2012	:.
1523	020212	004042	4042	:<
1524	020214	000012	12	:=
1525	020216	001012	1012	:>
1526	020220	001006	1006	:?
1527	020222	000042	42	:@
1528	020224	004400	4400	:A
1529	020226	004200	4200	:B
1530	020230	004100	4100	:C
1531	020232	004040	4040	:D
1532	020234	004020	4020	:E
1533	020236	004010	4010	:F
1534	020240	004004	4004	:G
1535	020242	004002	4002	:H
1536	020244	004001	4001	:I
1537	020246	002400	2400	:J
1538	020250	002200	2200	:K
1539	020252	002100	2100	:L
1540	020254	002040	2040	:M
1541	020256	002020	2020	:N
1542	020260	002010	2010	:O
1543	020262	002004	2004	:P
1544	020264	002002	2002	:Q
1545	020266	002001	2001	:R
1546	020270	001200	1200	:S
1547	020272	001100	1100	:T
1548	020274	001040	1040	:U
1549	020276	001020	1020	:V
1550	020300	001010	1010	:W
1551	020302	001004	1004	:X
1552	020304	001002	1002	:Y
1553	020306	001001	1001	:Z
1554	020310	004202	4202	:[
1555	020312	002202	2202	:]

1556	020314	002006	2006	:	^
1557	020316	001022	1022	:	
1558					
1559	020320	000000	0		
1560	020322	000000	0		
1561	020324	000000	0		
1562	020326	000000	0		
1563	020330	000000	0		
1564	020332	000000	0		
1565	020334	000000	0		
1566	020336	000000	0		
1567	020340	000000	0		
1568	020342	000000	0		
1569	020344	000000	0		
1570	020346	000000	0		
1571	020350	000000	0		
1572	020352	000000	0		
1573	020354	000000	0		
1574	020356	000000	0		
1575	020360	000000	0		
1576	020362	000000	0		
1577	020364	000000	0		
1578	020366	000000	0		
1579	020370	000000	0		
1580	020372	000000	0		
1581	020374	000000	0		
1582	020376	000000	0		
1583	020400	000000	0		
1584	020402	000000	0		
1585	020404	004100	CARD01: 4100	:	C
1586	020406	004400	4400	:	A
1587	020410	002001	2001	:	R
1588	020412	004040	4040	:	D
1589	020414	000000	0	:	SP
1590	020416	001000	1000	:	O
1591	020420	000400	400	:	1
1592	020422	004100	CARD02: 4100	:	C
1593	020424	004400	4400	:	A
1594	020426	002001	2001	:	R
1595	020430	004040	4040	:	D
1596	020432	000000	0	:	SP
1597	020434	001000	1000	:	O
1598	020436	000200	200	:	2
1599	020440	004100	CARD03: 4100	:	C
1600	020442	004400	4400	:	A
1601	020444	002001	2001	:	R
1602	020446	004040	4040	:	D
1603	020450	000000	0	:	SP
1604	020452	001000	1000	:	O
1605	020454	000100	100	:	3
1606	020456	004100	CARD04: 4100	:	C
1607	020460	004400	4400	:	A
1608	020462	002001	2001	:	R
1609	020464	004040	4040	:	D
1610	020466	000000	0	:	SP
1611	020470	001000	1000	:	O

1612	020472	000040			
1613	020474	004100	CARD05:	40	:4
1614	020476	004400		4100	:C
1615	020500	002001		4400	:A
1616	020502	004040		2001	:R
1617	020504	000000		4040	:D
1618	020506	001000		0	:SP
1619	020510	000020		1000	:0
1620	020512	004100	CARD06:	20	:5
1621	020514	004400		4100	:C
1622	020516	002001		4400	:A
1623	020520	004040		2001	:R
1624	020522	000000		4040	:D
1625	020524	001000		0	:SP
1626	020526	000010		1000	:0
1627	020530	004100	CARD07:	10	:6
1628	020532	004400		4100	:C
1629	020534	002001		4400	:A
1630	020536	004040		2001	:R
1631	020540	000000		4040	:D
1632	020542	001000		0	:SP
1633	020544	000004		1000	:0
1634	020546	004100	CARD08:	4	:7
1635	020550	004400		4100	:C
1636	020552	002001		4400	:A
1637	020554	004040		2001	:R
1638	020556	000000		4040	:D
1639	020560	001000		0	:SP
1640	020562	000002		1000	:0
1641	020564	004100	CARD09:	2	:8
1642	020566	004400		4100	:C
1643	020570	002001		4400	:A
1644	020572	004040		2001	:R
1645	020574	000000		4040	:D
1646	020576	001000		0	:SP
1647	020600	000001		1000	:0
1648	020602	004100	CARD10:	1	:9
1649	020604	004400		4100	:C
1650	020606	002001		4400	:A
1651	020610	004040		2001	:R
1652	020612	000000		4040	:D
1653	020614	000400		0	:SP
1654	020616	001000		400	:1
1655	020620	004100	CARD11:	1000	:0
1656	020622	004400		4100	:C
1657	020624	002001		4400	:A
1658	020626	004040		2001	:R
1659	020630	000000		4040	:D
1660	020632	000400		0	:SP
1661	020634	000400		400	:1
1662	020636	004100	CARD12:	400	:1
1663	020640	004400		4100	:C
1664	020642	002001		4400	:A
1665	020644	004040		2001	:R
1666	020646	000000		4040	:D
1667	020650	000400		0	:SP
				400	:1

1668	020652	000200			
1669	020654	004100			
1670	020656	004400	CARD13:	200	:2
1671	020660	002001		4100	:C
1672	020662	004040		4400	:A
1673	020664	000000		2001	:R
1674	020666	000400		4040	:D
1675	020670	000100		0	:SP
1676	020672	004100		400	:1
1677	020674	004400	CARD14:	100	:3
1678	020676	002001		4100	:C
1679	020700	004040		4400	:A
1680	020702	000000		2001	:R
1681	020704	000400		4040	:D
1682	020706	000040		0	:SP
1683	020710	004100		400	:1
1684	020712	004400	CARD15:	40	:4
1685	020714	002001		4100	:C
1686	020716	004040		4400	:A
1687	020720	000000		2001	:R
1688	020722	000400		4040	:D
1689	020724	000020		0	:SP
1690	020726	004100		400	:1
1691	020730	004400	CARD16:	20	:5
1692	020732	002001		4100	:C
1693	020734	004040		4400	:A
1694	020736	000000		2001	:R
1695	020740	000400		4040	:D
1696	020742	000010		0	:SP
1697	020744	004100		400	:1
1698	020746	004400	CARD17:	10	:6
1699	020750	002001		4100	:C
1700	020752	004040		4400	:A
1701	020754	000000		2001	:R
1702	020756	000400		4040	:D
1703	020760	000004		0	:SP
1704	020762	004100	CARD18:	400	:1
1705	020764	004400		4	:7
1706	020766	002001		4100	:C
1707	020770	004040		4400	:A
1708	020772	000000		2001	:R
1709	020774	000400		4040	:D
1710	020776	000002		0	:SP
1711	021000	004100		400	:1
1712	021002	004400	CARD19:	2	:8
1713	021004	002001		4100	:C
1714	021006	004040		4400	:A
1715	021010	000000		2001	:R
1716	021012	000400		4040	:D
1717	021014	000001		0	:SP
1718	021016	004100		400	:1
1719	021020	004400	CARD20:	1	:9
1720	021022	002001		4100	:C
1721	021024	004040		4400	:A
1722	021026	000000		2001	:R
1723	021030	000200		4040	:D
				0	:SP
				200	:2

1724 021032 001000  
1725  
1726  
1727  
1728  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736 021034 000000  
1737 021036 000000  
1738 021040 000000  
1739 021042 000000  
1740 021044 000000  
1741 021046 000000  
1742 021050 000000  
1743 021052 000000  
1744 021054 000000  
1745 021056 000000  
1746 021060 000000  
1747 021062 000000  
1748 021064 000000  
1749 021066 000000  
1750 021070 000000  
1751 021072 000000  
1752 021074 000000  
1753 021076 000000  
1754  
1755  
1756  
1757  
1758 021100 177160  
1759 021102 177161  
1760 021104 177162  
1761 021106 177163  
1762 021110 177164  
1763 021112 177165  
1764 021114 177166  
1765 021116 177167  
1766  
1767  
1768  
1769 021120 000230  
1770 021122 000232  
1771  
1772  
1773  
1774 021124 000004  
1775  
1776 000001

1000 :0

.EVEN  
ADR: XX  
BAD: XX  
CARD: XX  
CNT: XX  
CNT1: XX  
GOOD: XX  
HOLD: XX  
INPTR: XX  
LASTPC: XX  
LEVEL: XX  
LOC: XX  
OUTPTR: XX  
PT: XX  
PTR: XX  
SCOPE: XX  
TEMP: XX  
MPSAV: XX  
TOG1: XX

:  
:\*\*\*\*\* DEVICE ADDRESSES \*\*\*\*\*  
:

TCR: 177160  
TCRHI: 177161  
TSR: 177162  
TSRHI: 177163  
TDR: 177164  
TDRHI: 177165  
TIR: 177166  
TIRHI: 177167

:  
:\*\*\*\*\* INTERRUPT VECTOR ADDRESSES \*\*\*\*\*  
:

PCV: 230  
PSV: 232

:  
:\*\*\*\*\* BUS REQUEST LEVEL \*\*\*\*\*  
:

BRLV: 4

.END





ERRVEC=	000004	324#	364	365*	376*									
E1	013576	1145	1150#											
E2	013616	1151	1156#											
E3	013636	1137	1157	1161#										
E4	013660	1164	1166#											
FILLO	014741	1333*	1337*	1347	1384#									
G	= 000002	244#												
GB	= 000003	243#	394	395	396	405	411	435	436	437	463	466	476	477
		480	483	484	497	500	501	525	526	531	532	543	548	573
		574	598	624	627	633	634	648	649	654	656	663	664	668
		670	677	678	685	686	690	691	713	714	720	724	725	726
		737	738	742	743	758	759	784	785	798	799	800	811	812
		818	819	848	849	865	866	867	882	883	890	891	917	921
		1032	1037	1041										
GMSG	015622	1147	1438#											
GOOD	021046	394*	395*	396*	405*	411*	435*	436*	437*	459*	461	466*	476*	477*
		480*	483*	484*	497*	500*	501*	525*	526*	531*	532*	543*	548*	573*
		574*	598*	624*	627*	633*	634*	648*	649*	654*	656*	663*	664*	668*
		670*	677*	678*	685*	686*	690*	691*	713*	714*	720*	724*	725*	726*
		737*	738*	742*	743*	758*	759*	784*	785*	798*	799*	800*	811*	812*
		818*	819*	848*	849*	865*	866*	867*	882*	883*	890*	891*	913*	915
		921*	1028*	1030	1149	1741#								
GTSWR	014226	345	1250#											
HOLD	021050	403*	404	405	1742#									
INIT	001222	352	386#											
INITM1	015475	389	1435#											
INIT1	001242	393#	394	395	396									
INPTR	021052	778*	782*	796*	842*	846*	863*	880*	1022	1743#				
INTAG	000713	334#	1274											
K10	014310	1261	1265#											
K11	014332	1268	1270#											
K14	014336	1273#												
K15	014360	1240	1242	1246	1248	1275	1277#							
K16	014362	1266	1278#											
K17	014422	1285	1289#											
K18	014434	1280	1282	1292#										
K19	014246	1254#	1264											
K20	014302	1263#	1293											
K7	014252	1256#	1257	1291										
K9	014270	1260#												
LASTPC	021054	394	395	396	405	411	435	436	437	455	464	466	476	477
		480	483	484	494	497	500	501	525	526	531	532	543	548
		564	573	574	598	624	627	633	634	644	648	649	654	656
		663	664	668	670	677	678	685	686	690	691	713	714	720
		724	725	726	737	738	742	743	753	758	759	784	785	798
		799	800	811	812	818	819	848	849	865	866	867	882	883
		890	891	921	1135*	1744#								
		444*	452*	460	1745#									
LEVEL	021056	1304#	1396											
LF	014513	1415*	1416*	1746#										
LOC	021060	387	1434#											
LOGTST	015441	394#	395#	396#	405#	411#	435#	436#	437#	466#	476#	477#	480#	483#
LP	= 012320	484#	497#	500#	501#	525#	526#	531#	532#	543#	548#	573#	574#	598#
		624#	627#	633#	634#	648#	649#	654#	656#	663#	664#	668#	670#	677#
		678#	685#	686#	690#	691#	713#	714#	720#	724#	725#	726#	737#	738#
		742#	743#	758#	759#	784#	785#	798#	799#	800#	811#	812#	818#	819#



CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0056

		848#	849#	865#	866#	867#	882#	883#	890#	891#	921#
MNEW	014500	1253	1300#								
MODEO	014742	1334*	1338*	1343	1346*	1357*	1385#				
MP	= 000002	1019#	1025								
MPSAV	021074	1027*	1033	1043*	1752#						
MSWR	014470	1250	1298#								
OCTDEC	0:3144	1052#	1160								
OD1	013160	1055#	1059	1064							
OD2	013204	1057	1061#								
OD3	013222	1060	1065#								
ONES	013254	1053*	1055*	1056	1061*	1068	1072#				
OUTPTR	021062	853*	1003	1005*	1747#						
PASS1	004364	356	608#								
PASS2	006366	693	700#								
PASS3	007726	761	768#								
PASS4	011030	820	827#								
PCMSG	015576	1139	1437#								
PCV	021120	445*	487*	637*	746*	1085	1113	1119*	1769#		
PROGNM	017745	344	1466#								
PS	= 177776	225#	339*	443*	449*	489*	492*	639*	642*	748*	751*
PSV	021122	446*	488*	638*	747*	1121*	1770#				
PT	021064	1195*	1196	1748#							
PTR	021066	716*	722*	733*	942	1022*	1028	1045*	1749#		
P1A	004756	639#	644								
P1B	005032	645	647#								
P1D	005170	652#	658								
P1E	005666	673#	680								
P1INT	005026	637	646#								
P1MSG1	016623	612	153#								
P1MSG2	016705	618	1454#								
P1MSG3	016757	620	1455#								
P2A	006604	718#	728								
P2B	007526	752#	753								
P2C	007562	754	756#								
P2INT	007556	746	755#								
P2MSG1	017050	701	1456#								
P2MSG2	017136	703	773	835	1457#						
P2MSG3	017210	528	545	705	775	837	1458#				
P2MSG4	017274	391	1459#								
P2SD1	012450	945#	946	950							
P2SEND	012434	721	732	756	942#						
P3A	010160	790#	802								
P3MSG1	017334	769	1460#								
P3MSG2	017422	771	1461#								
P3SD1	012526	961#	962	966							
P3SD2	012554	967#	968	972							
P3SEND	012510	792	806	816	958#						
P4A	011300	854#	869								
P4MSG1	017510	828	1462#								
P4MSG2	017577	831	1463#								
P4MSG3	017645	833	1464#								
P4SD1	012704	1001#	1002	1008							
P4SD2	012744	1009#	1010	1014							
P4SEND	012674	857	877	888	999#						
P5SD1	012616	980#	981	985							
P5SD2	012644	986#	987	991							

PSSEND	012604	856	876	887	978#												
QUES	014511	1292	1302#														
RDOCT	014070	1108	1118	1125	1220#												
RD.1	014100	1222#	1231														
RT1	001150	365	372#														
RT2	001156	371	372	374#													
RT3	001172	369	376#														
RT4	001220	379	381#														
SB1	012420	935#	937														
SCOPE	021070	341*	394	395	396	405	411	435	436	437	455	464	466	476			
		477	480	483	484	494	497	500	501	525	526	531	532	543			
		548	564	573	574	598	624	627	633	634	644	648	649	654			
		656	663	664	668	670	677	678	685	686	690	691	713	714			
		720	724	725	726	737	738	742	743	753	758	759	784	785			
		798	799	800	811	812	818	819	848	849	865	866	867	882			
		883	890	891	921	1134*	1750#										
		839	934#														
SETBFS	012414	839	934#														
SR	= 177570	226#															
START	001000	337	339#														
ST1	001046	347	350#	361													
ST2	001062	351	354#														
ST3	001076	355	358#														
SWR	000700	328#	346	350	354	366*	368	374*	380*	1132	1136	1163	1239	1269*			
		1406															
SWRCK	001110	342	364#														
SWREG	000176	320#	374	1239	1251												
S78	014070	1217#															
TCR	021100	394	404*	405	409*	411	435	448*	466	477	480	484	486*	497			
		499*	500	526	532	554*	574	588*	624	626*	627	636*	649	652*			
		656	660*	664	666*	670	673*	676	678	682*	686	688*	689	691			
		709*	712	714	718*	723	725	730*	736	738	740*	743	745*	757			
		759	779*	783	785	790*	797	798	804*	810	812	814*	817	819			
		843*	847	849	854*	864	865	871*	881	883	885*	889	891	1081			
		1110	1758#														
TCRHI	021102	475*	482*	513*	534*	616*	1759#										
TDR	021110	914	947*	963*	982*	1003*	1011*	1029	1762#								
TDRHI	021112	969*	988*	1763#													
TEMP	021072	1198*	1199*	1200*	1201	1221*	1225*	1226*	1227*	1228*	1229*	1232	1751#				
TENS	013256	1054*	1062*	1065	1073#												
TEST1	001446	398	402#														
TEST2	001664	413	431#														
TEST3	002076	439	443#	455	464												
TEST4	002342	468	475#	476	477												
TEST5	003210	503	509#														
TEST6	004222	580	584#														
TIR	021114	396	437	634	653	661	667	674	683	710	726	734	742	780			
		794	800	808	844	859	860	867	878	911	921	1023	1764#				
TIRHI	021116	719	731	741	791	805	815	855	872	873	886	945	951	961			
		967	980	986	1001	1009	1765#										
TKB	= 177562	228#	1243	1258	1425												
TKS	= 177560	227#	1241	1256	1276*	1422*	1423										
TM1	013710	1180#	1186														
TM2	013726	1183	1185#														
TM3	013734	1181	1187#														
TOG	013260	1052*	1058*	1063*	1074#												
TOG1	021076	958*	965*	978*	984*	999*	1006*	1007	1012*	1013	1753#						





.MAIN. MACY11 30A(1052) 20-MAY-80 09:53 PAGE 6  
CZCTBA.P11 20-MAY-80 08:54

CROSS REFERENCE TABLE -- MACRO NAMES

SEQ 0060

CKTCR	271#	394	405	411	435	466	477	480	484	497	500	526	532	574	624
	627	649	656	664	670	678	686	691	714	725	738	743	759	785	798
	812	819	849	865	883	891									
CKTDR	291#														
CKTIR	301#	396	437	634	726	742	800	867	921						
CKTSR	281#	395	436	476	483	501	525	531	543	548	573	598	633	648	654
	663	668	677	685	690	713	720	724	737	758	784	799	811	818	848
	866	882	890												
LOOP	261#	394	395	396	405	411	435	436	437	455	464	466	476	477	480
	483	484	494	497	500	501	525	526	531	532	543	548	564	573	574
	598	624	627	633	634	644	648	649	654	656	663	664	668	670	677
	678	685	686	690	691	713	714	720	724	725	726	737	738	742	743
	753	758	759	784	785	798	799	800	811	812	818	819	848	849	865
	866	867	882	883	890	891	921								
WTDONE	247#	676	689	712	723	736	757	783	797	810	817	847	864	881	889
WTIDR	257#	653	661	667	674	683	710	734	780	794	808	844	859	878	
WTODR	252#	719	731	741	791	805	815	855	872	886					

. ABS. 021126 000

ERRORS DETECTED: 0

CZCTBA,CZCTBA/CRF=CZCTBA.P11  
RUN-TIME: 10 21 2 SECONDS  
RUN-TIME RATIO: 65/34-1.8  
CORE USED: 8K (15 PAGES)