

11/23/24/73

KEF11-B CIS DIAG  
CJKDHB0

AH-S433B-MC

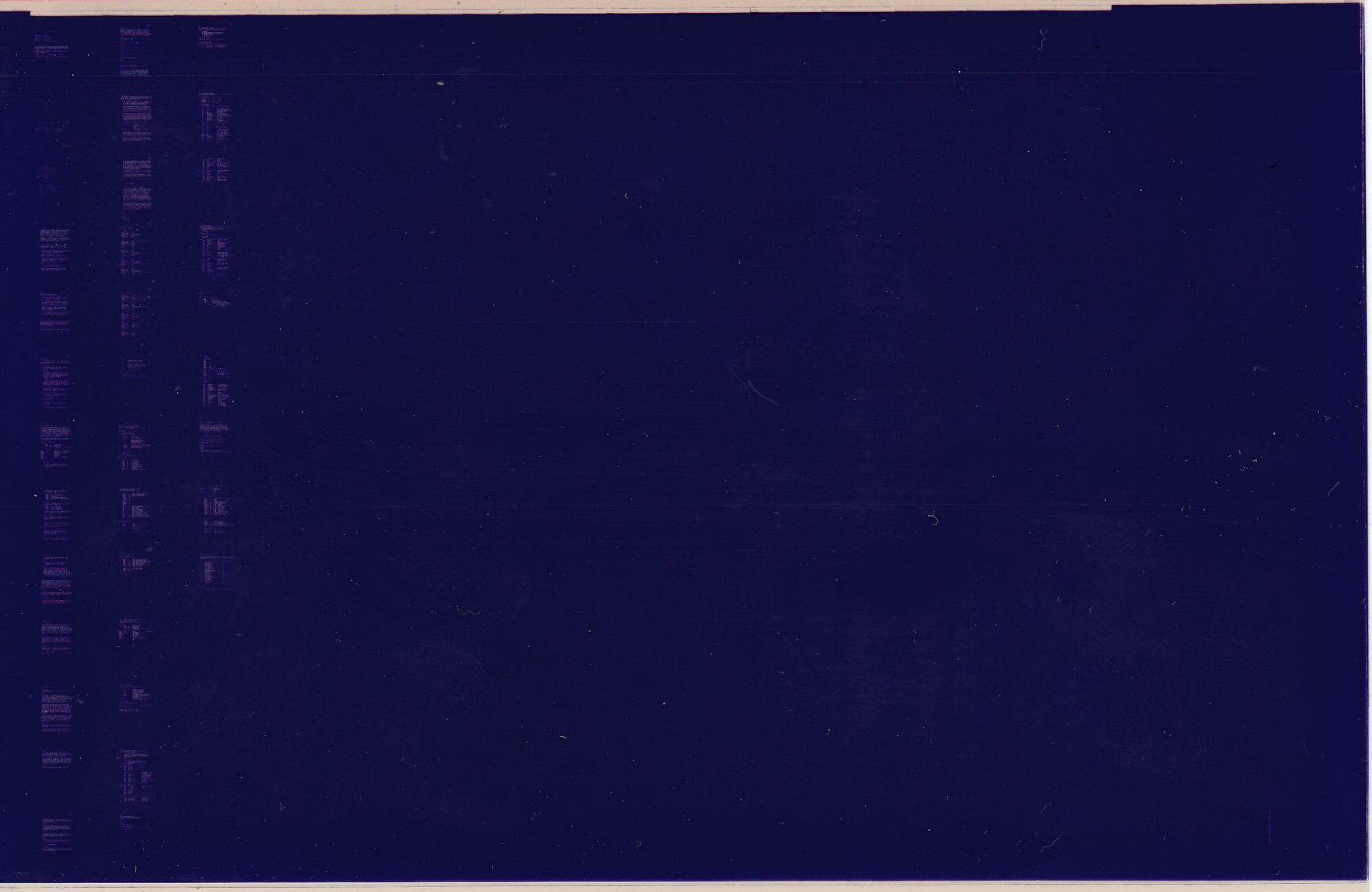
1 OF 2 JUL 1985

**digital**

COPYRIGHT © 1981-85

MADE IN USA

MICRO 11



11/23/24/73

MICRO 11

KEF11-B CIS DIAG  
CJKDHB0

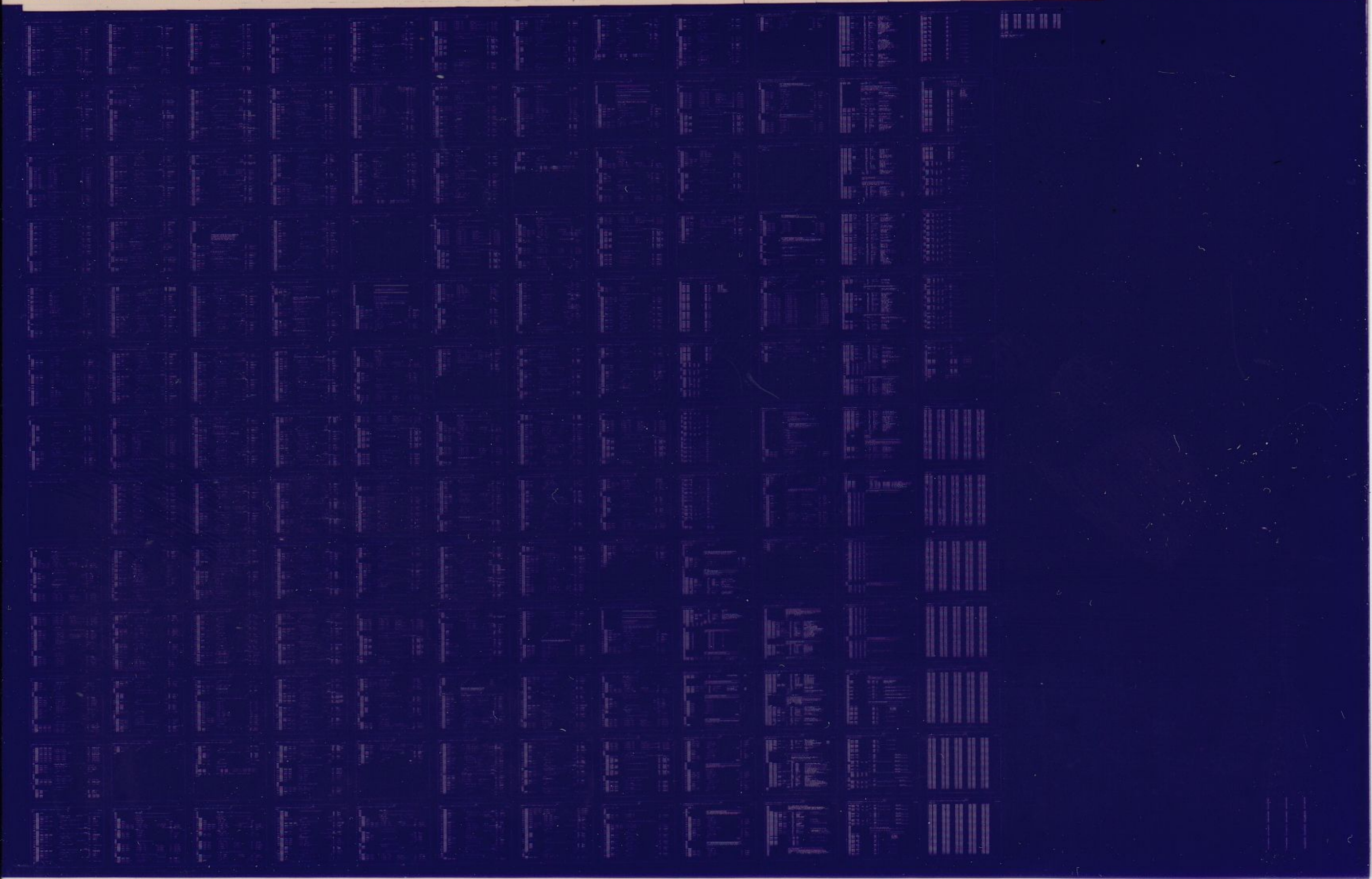
AH-S433B-MC

2 OF 2 JUL 1985

COPYRIGHT © 1981-85

**digital**

MADE IN USA



[ B w  
A U;  
1

SEQ 000

CJKDH80 KEF11-B CIS DIAGNOSTIC MACRO M1200 09-APR-85 11:22 PAGE 3

1  
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

.REM \*

IDENTIFICATION  
-----

PRODUCT CODE: AC-S431B-MC  
PRODUCT NAME: CJKDH80 KEF11-B CIS DIAG.  
PRODUCT DATE: APRIL 1985  
MAINTAINER: DIAGNOSTIC ENGINEERING

THE INFORMATION IN THIS DOCUMENT IS SUBJECT 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.

NO RESPONSIBILITY IS ASSUMED FOR THE USE OR RELIABILITY OF SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL OR ITS AFFILIATED COMPANIES.

COPYRIGHT (C): 1981, 1985 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	DECX/11

50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60

HISTORY

REVISION

DATE

REASONS/CHANGES

-----  
A  
B

-----  
JUNE 1981  
APRIL 1985

-----  
FIRST RELEASE  
MAKE UFD COMPATABLE

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

TABLE OF CONTENTS

-----

1.0	GENERAL PROGRAM INFORMATION
1.1	ABSTRACT
1.2	SYSTEM REQUIREMENTS
1.3	RELATED DOCUMENTS AND STANDARDS
1.4	DIAGNOSTIC HIERARCHY PREREQUISITES
1.5	ASSUMPTIONS
2.0	OPERATING INSTRUCTIONS
2.1	LOADING AND STARTING PROCEDURE
2.2	PROGRAM OPTIONS
2.3	EXECUTION TIMES
3.0	ERROR INFORMATION
3.1	ERROR REPORTING PROCEDURES
3.2	ERROR HALTS
4.0	PERFORMANCE AND PROGRESS REPORTS
4.1	PERFORMANCE REPORTS
4.2	PROGRESS REPORTS
5.0	DEVICE INFORMATION TABLES
6.0	PROGRAM DESCRIPTION
6.1	PROGRAM EXECUTION CHARACTERISTICS
6.2	SUBTEST SUMMARIES
6.3	SPECIAL SUBROUTINE DESCRIPTION
7.0	LISTING

97  
98  
99  
100  
101  
102  
103  
104  
105  
106  
107  
108  
109  
110  
111  
112  
113  
114  
115  
116  
117  
118  
119  
120  
121  
122  
123  
124  
125  
126  
127  
128  
129  
130  
131  
132  
133  
134  
135  
136  
137  
138  
139  
140  
141  
142  
143  
144  
145  
146  
147  
148  
149  
150  
151

## 1.0 GENERAL PROGRAM INFORMATION

### 1.1 ABSTRACT

THIS DIAGNOSTIC VERIFIES THE FUNCTIONALITY OF THE KEF11-BA (CIS COMMERCIAL INSTRUCTIONS SET) OPTION WITH BOTH MEMORY MANAGEMENT ENABLED AND DISABLED. IT ALLOWS THE USER TO CHECK OUT ANY COMBINATION OF CIS CHIPS. IN VIRTUALLY ALL CASES, FAULT ISOLATION IS TO THE CIS CHIP LEVEL. THE DIAGNOSTIC CAN BE RUN UNDER XXDP, ACT, AND APT MONITORS. THE SOFTWARE SWITCH REGISTER (LOCATION 176) IS USED.

THE CIS OPTION CONSISTS OF SIX MOS/LSI CONTROL CHIPS CONTAINED IN THREE 40-PIN HYBRID CARRIERS. ALL CIS INSTRUCTIONS ARE CHIP PARTITIONED, I.E. ALL THE CODE FOR A PARTICULAR INSTRUCTION IS CONTAINED ON A PARTICULAR CONTROL CHIP. THE EXCEPTION IS THAT ALL CIS INSTRUCTIONS MUST PASS THROUGH THE FIRST CIS CHIP (CONTROL CHIP DC303-004) BEFORE REACHING THEIR DESTINATION CHIP.

### 1.2 SYSTEM REQUIREMENTS

#### A. HARDWARE REQUIREMENTS

THIS DIAGNOSTIC IS DESIGNED TO RUN ON A SYSTEM IMPLEMENTING A DCF11-AA CHIP SET AND A MEMORY MANAGEMENT UNIT. ALSO REQUIRED ARE A SERIAL LINE INTERFACE, A CONSOLE TERMINAL, AND 24K OF MEMORY. THE FIRST CIS CHIP (DC303-004) MUST BE INSTALLED. ANY (OR NONE) OF THE OTHER CIS CHIPS MAY BE PRESENT.

#### B. SOFTWARE REQUIREMENTS

THE SOFTWARE ENVIRONMENTS UNDER WHICH THIS DIAGNOSTIC WILL OPERATE ARE APT SCRIPT, APT STAND-ALONE, ACT Q.V., ACT AUTO-ACCEPT, ACT STAND-ALONE, XXDP CHAIN, XXDP STAND-ALONE, AND ABSOLUTE BINARY PAPER TAPE.

### 1.3 RELATED DOCUMENTS AND STANDARDS

THE FOLLOWING DOCUMENTS WERE USED OR REFERENCED DURING THE CREATION OF THIS DIAGNOSTIC:

DIAGNOSTIC ENGINEERING STANDARDS AND CONVENTIONS PROGRAMMING PRACTICES (DOC. NO. 175-003-009-02)  
ACT11/XXDP PROGRAMMING SPECIFICATION (AUTOCAT-11-QZAU8-B-D)  
BIC/BIN EXTENSION LIST  
STANDARD APT SYSTEM TO PDP-11 DIAGNOSTIC INTERFACE (APT/11-317-07-09)  
DEC STD 168: PDP11 EXTENDED INSTRUCTIONS  
CIS4.LIS

153  
154  
155  
156  
157  
158  
159  
160  
161  
162  
163  
164  
165  
166  
167  
168  
169  
170  
171  
172  
173  
174  
175  
176

CIS5.LIS  
CIS6.LIS  
CIS7.LIS  
CIS8.LIS  
CIS9.LIS  
CONTROL CHIP SPECIFICATION  
DATA CHIP SPECIFICATION  
SPMAC REFERENCE MANUAL

1.4 DIAGNOSTIC HIERARCHY PREREQUISITES

IT IS ASSUMED THAT THE CPU, MEMORY MANAGEMENT, MEMORY, SERIAL LINE INTERFACE, AND CONSOLE TERMINAL HAVE ALL PASSED THE PROPER DIAGNOSTICS.

1.5 ASSUMPTIONS

THE DIAGNOSTIC WILL ATTEMPT TO TEST ALL SIX CIS CHIPS UNLESS THE SOFTWARE SWR IS ALTERED TO INDICATE A SUBSET OF CIS CHIPS TO BE TESTED. SEE SECTION 2.2.

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  
219  
220  
221  
222  
223  
224  
225  
226  
227  
228

2.0 OPERATING INSTRUCTIONS

2.1 LOADING AND STARTING PROCEDURE

A. LOADING

USE STANDARD LOADING PROCEDURE FOR PDP-11 ABSOLUTE BINARY PAPER TAPES; OR XXDP MEDIA (FILES WITH .BIC OR .BIN EXTENSIONS ONLY).

B. STARTING

THE PROGRAM SHOULD ALWAYS BE STARTED AT 200. UPON START UP (RUNNING IN NON-AUTOMATIC MODE) THE DIAGNOSTIC INITIALIZES, PRINTS OUT ITS IDENTIFICATION (FIRST TIME ONLY), AND THEN PRINTS:

"SWR = 000000 NEW = "

THE NUMBER FOLLOWING "SWR" IS THE CURRENT VALUE OF THE SOFTWARE SWITCH REGISTER. IF NO CHANGES ARE DESIRED IN THE SWITCH REGISTER THEN A CARRIAGE RETURN MAY BE HIT; IF CHANGES ARE DESIRED THEN A NEW VALUE MAY BE TYPED IN FOLLOWED BY A CARRIAGE RETURN. IF NOT ALL SIX CIS CHIPS ARE TO BE TESTED THEN THE SWR WILL HAVE TO BE CHANGED. SEE NEXT SECTION FOR SWR BIT ASSIGNMENTS.

2.2 PROGRAM OPTIONS

SWITCH REGISTER SETTINGS

LOCATION 176 IS USED FOR THE SOFTWARE SWITCH REGISTER (UNLESS RUNNING UNDER APT SCRIPT MODE). THE FOLLOWING ARE THE RESULTS OF SETTING THE RESPECTIVE SWR BITS:

BIT15	-HALT ON ERROR
BIT14	-SCOPE LOOP
BIT13	-INHIBIT ERROR TYPEOUT
BIT12	-NOT USED
BIT11	-INHIBIT SUBTEST ITERATIONS
BITS<10:8>	-NOT USED
BIT07	-MEMORY MANAGEMENT ALWAYS DISABLED
BIT06	-MEMORY MANAGEMENT ALWAYS ENABLED
BIT05	-TEST CIS CONTROL CHIP 9 (DC303-009)
BIT04	-TEST CIS CONTROL CHIP 8 (DC303-008)
BIT03	-TEST CIS CONTROL CHIP 7 (DC303-007)
BIT02	-TEST CIS CONTROL CHIP 6 (DC303-006)
BIT01	-TEST CIS CONTROL CHIP 5 (DC303-005)
BIT00	TEST CIS CONTROL CHIP 4 (DC303 004)



230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
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

AS CAN BE SEEN ABOVE, SWR BITS<5:0> ALLOW THE USER TO SELECT ANY COMBINATION OF CIS CONTROL CHIP(S) TO BE TESTED. TO SELECT A CIS CONTROL CHIP FOR TESTING, ITS CORRESPONDING BIT IN THE SWITCH REGISTER MUST BE SET. NOTE, HOWEVER, THAT IF SWR BITS<5:0> ARE ALL ZERO (DEFAULT MODE) THEN THE DIAGNOSTIC WILL TEST ALL SIX CIS

CONTROL CHIPS.

THE VALUE OF THE SOFTWARE SWITCH REGISTER MAY BE CHANGED DURING THE PROGRAM RUN. TO DO THIS A CONTROL/G IS TYPED (TYPE THE 'CTRL' AND 'G' KEYS SIMULTANEOUSLY); WITHIN A FEW SECONDS THE CURRENT VALUE OF THE SWR IS TYPED OUT AND THE PROGRAM WAITS FOR NEW INPUT.

NOTE: THE PROGRAM IS NOT RUNNING ANY TESTS WHILE WAITING FOR A NEW VALUE OF SWR TO BE INPUT.

2.3 EXECUTION TIMES

ALL TIMES WERE MEASURED ON A KDF11-A, RUNNING THE FULL COMPLEMENT OF CIS CONTROL CHIPS WITH MEMORY MANAGEMENT ENABLED AND DISABLED.

A. FIRST PASS (QV)

30 SECONDS

B. LONGEST TEST

50 SECONDS

C. ADDITIONAL TIME FOR UNITS

N/A

D. FULL PASS TIME (ITERATIONS)

90 SECONDS

275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323

### 3.0 ERROR INFORMATION

#### 3.1 ERROR REPORTING PROCEDURES

WHEN A CIS INSTRUCTION FAILS, THE TYPEOUT GIVES THE ERROR PC, THE CONTROL CHIP NUMBER, AND WHAT KIND OF FAILURE OCCURRED. THE ERROR PC IS THE ACTUAL LOCATION OF THE ERROR CALL IN THE FAILING SUBTEST. THE CONTROL CHIP NUMBER REFERS TO WHICH CONTROL CHIP WAS BEING TESTED WHEN THE ERROR OCCURRED. THE CONTROL CHIP NUMBER WILL APPEAR AS 23-\*04, 23-\*05, 23 \*06, 23 \*07, 23-\*08, OR 23-\*09. THE \* SIGNIFIES THE REVISION LEVEL OF THE CHIP AND SO MAY BE ANY DIGIT. THE LAST TWO DIGITS DISTINGUISHES ONE CIS CHIP FROM ANOTHER. CURRENTLY, THE SIX CIS CONTROL CHIPS ARE PACKAGED ON THREE HYBRID CARRIERS. THE PART NUMBERS FOR THE HYBRID ASSEMBLIES ARE: 5700002-00 (CIS CHIPS 23-\*04 AND 23-\*06), 5700003-00 (CIS CHIPS 23-\*05 AND 23-\*07) AND 5700004-00 (CIS CHIPS 23-\*08 AND 23-\*09). THE KINDS OF FAILURES REPORTED ARE:

"CIS INSTRUCTION FAILURE"

"CIS INTERRUPT FAILURE"

"NO TRAP ON ILLEGAL CIS CLASS OPCODE"

THE "CIS INSTRUCTION FAILURE" IS THE MOST FREQUENTLY CALLED ERROR. IT USUALLY REFERS TO A CIS INSTRUCTION THAT PRODUCES THE WRONG CONDITION CODES, THE WRONG DESTINATION OUTPUT, OR THE WRONG GENERAL PURPOSE REGISTER CONTENTS. TO DETERMINE WHAT THE SPECIFIC FAILING INSTRUCTION IS, AND THE SPECIFIC TYPE OF FAILURE, LOOK IN THE DIAGNOSTIC AT THE CODE IN THE AREA OF THE ERROR PC.

#### 3.2 ERROR HALTS

THREE UNEXPECTED FAULTS CAN ALSO OCCUR. THEY ARE:

"AN UNEXPECTED TIME-OUT TRAP"

"AN UNEXPECTED ILLEGAL AND RESERVED INSTRUCTION TRAP"

"POWER FAIL"

IN ALL THREE CASES THE PROGRAM HALTS IMMEDIATELY AFTER TYPING OUT THE APPROPRIATE MESSAGE. HITTING "P" (PROCEED) WILL CAUSE THE DIAGNOSTIC TO START OVER IF THE FAULT WAS A POWER FAIL, AND TO CONTINUE THE DIAGNOSTIC FROM THE POINT AT WHICH IT WAS INTERRUPTED FOR THE OTHER TWO FAULTS. ANY OTHER UNEXPECTED INTERRUPTS OR TRAPS WILL BE CAUGHT BY THE TRAP CATCHER, WHICH ALSO HALTS THE DIAGNOSTIC.

SETTING SWR BIT15 WILL CAUSE THE THREE ERRORS LISTED IN SECTION 3.1 TO HALT AFTER TYPING OUT THE ERROR MESSAGE. HITTING "P" (PROCEED) WILL CONTINUE THE DIAGNOSTIC.

325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338

4.0 PERFORMANCE AND PROGRESS REPORTS

4.1 PERFORMANCE REPORTS

N/A

4.2 PROGRESS REPORTS

AT THE END OF EACH PASS THE DIAGNOSTIC PRINTS OUT "END PASS #"  
FOLLOWED BY THE CURRENT PASS NUMBER.

340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390

## 5.0 DEVICE INFORMATION TABLES

THE CIS INSTRUCTIONS ARE PARTITIONED SUCH THAT NEARLY ALL THE MICRO-CODE FOR AN INSTRUCTION IS LOCATED ON ONE CHIP. (THE INSTRUCTIONS THAT RESIDE ON CONTROL CHIP DC303-004 ARE CONTAINED ENTIRELY ON THAT CHIP; ALL THE OTHER INSTRUCTIONS MUST FIRST PASS THROUGH CONTROL CHIP 4 SO THERE IS A SMALL AMOUNT OF MICRO-CODE FOR THE OTHER INSTRUCTIONS ON THAT CHIP.) THE PARTITIONING IS AS FOLLOWS:

CONTROL CHIP DC303-004:		
L2DR	07602R	LOAD 2 DESCRIPTORS
L3DR	07606R	LOAD 3 DESCRIPTORS
MOVC	076030	MOVE CHARACTER
MOVCR	076031	MOVE REVERSE JUSTIFIED CHARACTER
MOVTC	076032	MOVE TRANSLATED CHARACTER
LOCC	076040	LOCATE CHARACTER
SKPC	076041	SKIP CHARACTER
SCANC	076042	SCAN CHARACTER
SPANC	076043	SPAN CHARACTER
CMPC	076044	COMPARE CHARACTER
MATC	076045	MATCH CHARACTER
CONTROL CHIP DC303-005:		
CVTNL	076053	CONVERT NUMERIC TO LONG
CVTPN	076054	CONVERT PACKED TO NUMERIC
CVTNP	076055	CONVERT NUMERIC TO PACKED
CVTLN	076057	CONVERT LONG TO NUMERIC
CVTPL	076073	CONVERT PACKED TO LONG
CVTLP	076077	CONVERT LONG TO PACKED
CONTROL CHIP DC303-006:		
ADDN	076050	ADD NUMERIC
SUBN	076051	SUBTRACT NUMERIC
CMPN	076052	COMPARE NUMERIC
ASHN	076056	ARITHMETIC SHIFT NUMERIC
CONTROL CHIP DC303-007:		
ADDP	076070	ADD PACKED
SUBP	076071	SUBTRACT PACKED
CMPP	076072	COMPARE PACKED
ASHP	076076	ARITHMETIC SHIFT PACKED
CONTROL CHIP DC303-008:		
MULP	076074	MULTIPLY PACKED
CONTROL CHIP DC303-009:		
DIVP	076075	DIVIDE PACKED

ALL OF THE ABOVE INSTRUCTIONS EXCEPT FOR L2DR AND L3DR HAVE AN IN-LINE VERSION, TOO. THE MNEMONIC IS THE SAME EXCEPT AN "I" IS APPENDED TO IT AND THE OPCODE IS THE SAME EXCEPT BIT06 IS SET. FOR EXAMPLE, THE IN-LINE VERSION OF CVTNP (076055) IS CVTNPI (076155).

392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447

## 6.0 PROGRAM DESCRIPTION

THIS PROGRAM WAS WRITTEN IN MODULES. THE FIRST MODULE INCLUDES START UP AND INITIALIZATION ROUTINES, BASIC DEFINITIONS, VARIABLE DEFINITIONS, THE TRAP CATCHER, AND ACT AND APT INFORMATION. THE SECOND MODULE CONTAINS THE TESTS FOR THE CONTROL CHIP DC303-004 INSTRUCTIONS, THE TEST DATA FOR THOSE INSTRUCTIONS, AN ILLEGAL OPCODE TEST, AND AN INTERRUPTABILITY TEST. THE THIRD MODULE CONTAINS THE TESTS AND TEST DATA FOR CONTROL CHIP DC303-005 INSTRUCTIONS. A SUBROUTINE TO CONVERT LONG INTEGER TO AN INTERMEDIATE FORM, AND A SUBROUTINE TO CONVERT FROM THE INTERMEDIATE FORM TO LONG INTEGER. THE FOURTH MODULE CONTAINS THE TESTS FOR CONTROL CHIP DC303-006 INSTRUCTIONS. THE TESTS FOR CONTROL CHIP DC303-007 ARE IN THE FIFTH MODULE. THE SIXTH MODULE HAS THE TESTS AND TEST DATA FOR CONTROL CHIPS DC303-008 AND DC303-009. FINALLY, THE SEVENTH MODULE CONTAINS END OF PASS, SCOPE, ALL THE OTHER COMMON SUBROUTINES, ASCII MESSAGES, BUFFERS, TABLES, AND TEST DATA FOR CONTROL CHIPS DC303-006 AND DC303-007. WITH BITS 6 AND 7 OF THE SWR EQUAL TO 0 THE DIAGNOSTIC WILL EXECUTE EVERY INSTRUCTION WITH MEMORY MANAGEMENT ENABLED AND DISABLED.

### 6.1 PROGRAM EXECUTION CHARACTERISTICS

### 6.2 SUBTEST SUMMARIES

THERE IS ONE SUBTEST FOR EACH CIS INSTRUCTION. FIRST, THE REGISTER FORM OF THE INSTRUCTION IS TESTED WITH SEVERAL SETS OF DATA. THEN ONE CASE OF THE IN-LINE VERSION IS TESTED. THE TEST DATA FOR THE REGISTER FORM IS IN TABLES. THE FIRST WORD OF EACH TABLE IS THE NUMBER OF TEST CASES CONTAINED IN THE TABLE. THE FORMAT OF THE TEST DATA IS DIFFERENT FOR DIFFERENT INSTRUCTIONS; THE FIRST TEST CASE IN EACH TABLE HAS COMMENTS IDENTIFYING EACH ELEMENT IN THE TEST CASE.

THE FOLLOWING IS A SUMMARY OF EACH SUBTEST.

MOVC - THE TABLE OF TEST CASES CONTAINS:

<SOURCE LENGTH>  
<SOURCE ADDRESS>  
<DESTINATION LENGTH>  
<DESTINATION ADDRESS>  
<FILL CHARACTER>  
<SOURCE BUFFER STATE>  
<DESTINATION BUFFER STATE>

THE SOURCE AND DESTINATION BUFFER STATES DESIGNATE WHAT THOSE BUFFERS WILL BE FILLED WITH BEFORE EXECUTION OF THE MOVC INSTRUCTION. A '0' INDICATES EACH BYTE WILL CONTAIN 000, A '-1' INDICATES 377, AND A '1' INDICATES 252 (ALTERNATING 1'S AND 0'S BIT PATTERN). AN EXPECTED DESTINATION RESULT IS COMPUTED, AS WELL AS EXPECTED CONDITION CODES AND VALUES OF R0 R5. THEN THE MOVC INSTRUCTION IS EXECUTED AND THE ACTUAL RESULTS ARE COMPARED WITH THE EXPECTED RESULTS. DISAGREEMENTS ARE

449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502

FLAGGED WITH AN ERROR CALL, WHICH PRINTS OUT 'CIS INSTRUCTION FAILURE', THE PC OF THE ERROR CALL AND THE CONTROL CHIP NUMBER (IN THIS CASE, CONTROL CHIP NUMBER DC303-004). THIS WHOLE PROCEDURE IS REPEATED FOR EACH TEST CASE IN THE TABLE. THEN ONE SIMPLE IN-LINE CASE IS TESTED. THE DATA FOR THE IN-LINE CASE FOLLOWS THE TABLE OF TEST CASES FOR THE REGISTER FORM OF MOVCL, AND IT CONTAINS ONLY: SOURCE LENGTH, SOURCE ADDRESS, DESTINATION LENGTH, AND DESTINATION ADDRESS.

MOVRC - SAME AS MOVCL

MOVTC - SAME AS MOVCL,  
EXCEPT THAT A TRANSLATION TABLE, LOCATED AT 'TRANS', IS USED. PRIOR TO THE EXECUTION OF THE MOVTC INSTRUCTION, THE TRANSLATION TABLE IS CLEARED. THEN BYTE LOCATION TRANS+0 GETS A 63, TRANS+252 GETS A 314, AND TRANS+377 GETS A 125 (ALL NUMBERS OCTAL).

LOCC - THE TABLE OF TEST CASES CONTAINS:

<SOURCE LENGTH>  
<CHARACTER SEARCHED FOR>  
<OFFSET>

THE SEARCH BUFFER (BUFF1) IS FILLED WITH ZEROES, THEN AT BYTE LOCATION 'BUFF1+OFFSET' THE 'CHARACTER SEARCHED FOR' IS INSERTED. EXPECTED RESULTS FOR THE CONDITION CODES AND RO-R5 ARE COMPUTED. THE LOCC INSTRUCTION IS EXECUTED AND EXPECTED RESULTS ARE COMPARED WITH ACTUAL RESULTS. DISAGREEMENTS ARE FLAGGED WITH ERROR CALLS.

SKPC - SIMILAR TO LOCC.

EXCEPT THAT THE TEST CASES HAVE A 'NOT SEARCHED FOR CHARACTER' IN PLACE OF A 'CHARACTER SEARCHED FOR'. THE SEARCH BUFFER (BUFF1) IS FILLED WITH SEARCHED FOR CHARACTERS, THEN AT BYTE LOCATION 'BUFF1+OFFSET' THE 'NOT SEARCHED FOR' CHARACTER IS INSERTED.

SCANC - THE TABLE OF TEST CASES CONTAINS:

<SOURCE LENGTH>  
<OFFSET>

BUFF1 IS FILLED WITH ZEROES, THEN AT BYTE LOCATION 'BUFF1+OFFSET' A 125 IS INSERTED. THE TABLE 'CHRSET' IS CLEARED, THEN A 1 IS INSERTED AT BYTE LOCATION 'CHRSET+125', TO DENOTE MEMBERSHIP IN THE CHARACTER SET. EXPECTED CONDITION CODES AND RO-R5 VALUES ARE COMPUTED, THE SCANC INSTRUCTION IS EXECUTED, AND DISAGREEMENTS BETWEEN ACTUAL AND EXPECTED RESULTS CAUSE ERROR CALLS.

504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554

SPANC - THE TABLE OF TEST CASES CONTAINS:

<SOURCE LENGTH>

<OFFSET>

BUFF1 IS FILLED WITH ZEROES, THEN A 252 IS INSERTED AT BYTE LOCATION 'BUFF'.OFFSET'. CHRSET IS FILLED WITH ZEROES, THEN BYTE LOCATION 'CHRSET.0' GETS A 40 SO THAT A 0 IN THE SOURCE BUFFER WILL BE CONSIDERED A MEMBER OF THE CHARACTER SET AND A 252 WILL NOT. EXPECTED CONDITION CODES AND RO-R5 VALUES ARE COMPUTED, SPANC IS EXECUTED, EXPECTED RESULTS ARE COMPARED WITH ACTUAL RESULTS, AND DISAGREEMENTS CAUSE ERROR CALLS.

CMPC - THE TABLE OF TEST CASES CONTAINS:

<SOURCE1 LENGTH>

<SOURCE2 LENGTH>

BOTH SOURCE STRINGS ARE FILLED WITH 377'S AND THE FILL CHARACTER IS SET EQUAL TO 377. THEN IN THE LONGER STRING, AT 'SRC.ADR+OFFSET', A BYTE IS CLEARED. THIS BYTE WILL BE THE INEQUALITY BETWEEN THE TWO STRINGS WHICH WILL TERMINATE THE CMPC COMPARISON. ACTUAL RESULTS WILL BE COMPARED WITH PREVIOUSLY COMPUTED EXPECTED RESULTS (CONDITION CODES AND RO-R5 VALUES).

MATC - THE TABLE OF TEST CASES CONTAINS:

<OBJECT LENGTH>

<OFFSET>

THE SOURCE STRING IS THE FIRST 70 BYTES OF BUFF1. BUFF1 IS FIRST FILLED WITH 377'S AND THEN CLEARED FROM 'BUFF1.OFFSET' TO THE END OF BUFF1. THE OBJECT STRING STARTS AT BUFF2, IS OF LENGTH 'OBJECT LENGTH' FROM THE TABLE OF TEST CASES, AND IS ALL ZEROES. EXPECTED VALUES OF CONDITION CODES AND RO-R5 ARE COMPUTED, MATC IS EXECUTED, EXPECTED RESULTS AND ACTUAL RESULTS ARE COMPARED, AND ERROR CALLS ARE MADE FOR DISAGREEMENTS.

CVTLP - THE TABLE OF TEST CASES CONTAINS:

<DESTINATION DATA TYPE AND LENGTH>

<LONG INTEGER HIGH, AND SIGN>

<LONG INTEGER LOW>

THE LONG INTEGER IN THE TEST CASE IS CONVERTED TO PACKED AND PLACED INTO BUFFER 'DS.EXP' (DECIMAL STRING EXPECTED) ACCORDING TO THE DESTINATION TYPE AND LENGTH SPECIFIED IN THE TABLE. THE EXPECTED VALUES OF THE CONDITION CODES AND RO-R5 ARE ALSO COMPUTED. THE CVTLP INSTRUCTION IS EXECUTED AND EXPECTED RESULTS ARE COMPARED WITH ACTUAL RESULTS. DISAGREEMENTS WILL CAUSE ERROR CALLS.

556  
557  
558  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607

CVTLN - SAME AS THE CVTLP TEST,  
EXCEPT THE LONG INTEGER IS CONVERTED TO A NUMERIC DECIMAL  
FORM, ACCORDING TO THE SPECIFICATION IN THE TEST CASE.

CVTPL THE TABLE OF TEST CASES CONTAINS:

<SOURCE DATA TYPE AND LENGTH>  
<POINTER TO PACKED DECIMAL STRING>  
<A PACKED DECIMAL STRING>

THE PACKED DECIMAL STRING IS CONVERTED TO A LONG INTEGER  
AND THE RESULT IS PUT INTO 'R2.EXP' AND 'R3.EXP'  
(EXPECTED RESULTS FOR R2 AND R3). THE EXPECTED RESULTS  
FOR THE OTHER GENERAL PURPOSE REGISTER AND THE CONDITION  
CODES ARE ALSO COMPUTED. THE CVTPL INSTRUCTION IS  
EXECUTED, ACTUAL RESULTS ARE COMPARED WITH EXPECTED  
RESULTS, AND DISAGREEMENTS ARE FLAGGED BY ERROR CALLS.

CVTNL - SAME AS THE CVTNL TEST,

EXCEPT THAT NUMERIC DECIMAL STRINGS ARE CONVERTED INSTEAD  
OF PACKED DECIMAL STRINGS. ALSO, A TABLE FOR  
INTERPRETING ACCEPTABLE SIGN/DIGIT COMBINATIONS FOR  
OVERPUNCH IS INCLUDED.

CVTPN - THE TABLE OF TEST CASES CONTAINS:

<SOURCE DATA TYPE AND LENGTH>  
<POINTER TO A PACKED SOURCE STRING>  
<DESTINATION LENGTH>

EACH PACKED DECIMAL STRING IS CONVERTED INTO EACH OF THE  
SIX NUMERIC DECIMAL DATA TYPES. THUS, EACH TEST CASE IS  
USED SIX TIMES BEFORE GOING ON TO THE NEXT ONE. EACH  
TIME, THE EXPECTED CONDITION CODES AND RO-R5 VALUES ARE  
ALSO COMPUTED. THE CVTPN INSTRUCTION IS EXECUTED,  
EXPECTED RESULTS ARE COMPARED WITH ACTUAL RESULTS, AND  
DISAGREEMENTS ARE FLAGGED BY ERROR CALLS.

CVTNP - THE TABLE OF TEST CASES CONTAINS:

<SOURCE DATA TYPE AND LENGTH>  
<POINTER TO A SOURCE STRING>  
<DESTINATION DATA TYPE AND LENGTH>  
<A SOURCE STRING>

THE SOURCE STRING IS IN AN INTERMEDIATE FORMAT. THE  
FIRST WORD INDICATES THE SIGN (0 FOR POSITIVE, -1 FOR  
NEGATIVE). FOLLOWING THAT IS THE MAGNITUDE, ONE  
BINARY-CODED-DECIMAL DIGIT PER BYTE. THE SOURCE IS  
CONVERTED TO A PACKED DECIMAL STRING AND PUT INTO  
'DS.EXP', AND THE EXPECTED VALUES OF THE CONDITION CODES  
AND RO-R5 ARE COMPUTED. THE CVTNP INSTRUCTION IS  
EXECUTED, ACTUAL RESULTS COMPARED WITH EXPECTED RESULTS,  
AND DISAGREEMENTS FLAGGED BY ERROR CALLS.



609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663

ADDN - THE TABLE OF TEST CASES CONTAINS:

<SOURCE1 LENGTH>  
<SOURCE1 POINTER>  
<SOURCE2 LENGTH>  
<SOURCE2 POINTER>  
<DESTINATION LENGTH>  
<DESTINATION POINTER (SUM)>  
<EXPECTED CONDITION CODES>

THE SOURCE1, SOURCE2, AND DESTINATION STRINGS ARE IN AN INTERMEDIATE FORMAT. EACH ONE IS TESTED IN FIVE OF THE SIX DATA TYPE FORMATS (UNSIGNED ZONED IS TESTED IN THE INLINE CASE ONLY). THIS GIVES A TOTAL OF 125 TESTS PERFORMED ON EACH TEST CASE, IN ORDER TO TEST ALL COMBINATIONS OF THE FIVE DATA TYPES. SOURCE1 AND SOURCE2 ARE ADDED TOGETHER BY THE ADDN INSTRUCTION AND THE ANSWER IS COMPARED TO THE EXPECTED ANSWER WHICH IS SUPPLIED IN THE TEST TABLE (POINTED TO BY THE DESTINATION POINTER). EXPECTED CONDITION CODES ARE ALSO SUPPLIED IN THE TEST CASE TABLE AND IS COMPARED WITH THE ACTUAL CONDITION CODES. THE EXPECTED GENERAL PURPOSE REGISTER VALUES (R0-R5) ARE THE ONLY EXPECTED RESULTS THAT ARE ACTUALLY CALCULATED BY THE DIAGNOSTIC, AND THESE TOO ARE COMPARED WITH THE ACTUAL RESULTS. ANY DISAGREEMENTS BETWEEN ACTUAL AND EXPECTED RESULTS ARE FLAGGED BY AN ERROR CALL.

SUBN - THE SAME AS ADDN,

EXCEPT THAT THE TEST CASES HAVE THE EXPECTED DESTINATION STRING (IN AN INTERMEDIATE FORMAT) AND EXPECTED CONDITION CODES FOR A SUBTRACT OPERATION INSTEAD OF AN ADD OPERATION. THE SOURCE1 LENGTH AND POINTER, THE SOURCE2 LENGTH AND POINTER, AND THE DESTINATION LENGTH ARE IDENTICAL TO THOSE USED IN THE ADDN INSTRUCTION. THE TEST OPERATES IN EXACTLY THE SAME MANNER AS THE ADDN TEST.

CMPN - THE TABLE OF TEST CASES CONTAINS:

<SOURCE1 LENGTH>  
<SOURCE1 POINTER>  
<SOURCE2 LENGTH>  
<SOURCE2 POINTER>

THE TWO SOURCE STRINGS ARE IN AN INTERMEDIATE FORMAT. EACH ONE IS TESTED IN FIVE OF THE SIX NUMERIC DATA TYPE FORMATS (UNSIGNED ZONED IS TESTED IN THE INLINE CASE ONLY). THIS GIVES A TOTAL OF 25 TESTS PERFORMED ON EACH TEST CASE. IN EACH TEST, SOURCE1 IS ARITHMETICALLY COMPARED WITH SOURCE2 AND THE CONDITIONS ARE UPDATED TO REFLECT THE RESULTS. THE EXPECTED CONDITION CODES ARE COMPARED WITH THE ACTUAL CONDITION CODES, AND R0 R5 ARE TESTED TO SEE THAT THEY ARE CLEAR. ANY DISCREPANCIES ARE FLAGGED BY ERROR CALLS.

665  
666  
667  
668  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
700  
701  
702  
703  
704  
705

ASHN - THE TABLE OF TEST CASES CONTAINS:  
<SOURCE LENGTH>  
<DESTINATION LENGTH>  
<SHIFT COUNT>  
<ROUNDING DIGIT>

THE SOURCE STRING IS ALWAYS A STRING OF "4"'S STARTING AT DSBUF1 AND OF THE LENGTH SPECIFIED BY THE TEST CASE. EACH TIME, THE EXPECTED RESULTS ARE CALCULATED AND PUT INTO DS.EXP AND CC.EXP. ACTUAL RESULTS ARE COMPARED WITH EXPECTED RESULTS AND DISAGREEMENTS ARE FLAGGED BY ERROR CALLS.

ADDP - SAME AS THE ADDN TEST, EXCEPT THAT EACH TEST CASE IS RUN ONCE, USING SIGNED PACKED FORMAT. UNSIGNED PACKED IS TESTED IN THE IN LINE TEST. THE IDENTICAL TABLE OF TEST CASES USED FOR ADDN IS USED HERE.

SUBP - SAME AS THE SUBN TEST, EXCEPT THAT EACH TEST CASE IS RUN ONCE, USING SIGNED PACKED FORMAT. UNSIGNED PACKED IS TESTED IN THE IN LINE TEST. THE IDENTICAL TABLE OF TEST CASES USED FOR SUBN IS USED HERE.

CMPP - SAME AS THE CMPN TEST, EXCEPT THAT EACH TEST CASE IS RUN ONCE, USING SIGNED PACKED FORMAT. UNSIGNED PACKED IS TESTED IN THE IN-LINE TEST. THE IDENTICAL TABLE OF TEST CASES USED FOR CMPN IS USED HERE.

ASHP - SAME AS THE ASHN TEST, EXCEPT THAT EACH TEST CASE IS RUN ONCE, USING SIGNED PACKED FORMAT. UNSIGNED PACKED IS TESTED IN THE IN LINE TEST. THE IDENTICAL TABLE OF TEST CASES USED FOR ASHN IS USED HERE.

707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
740  
741

MULP THE TABLE OF TEST CASES CONTAINS:  
<SOURCE1 LENGTH>  
<SOURCE1 POINTER>  
<SOURCE2 LENGTH>  
<SOURCE2 POINTER>  
<DESTINATION LENGTH>  
<DESTINATION POINTER (PRODUCT)>  
<CC.EXP (PRODUCT)>

THE TWO SOURCES (MULTIPLIER AND MULTIPLICAND) AND EXPECTED DESTINATION (PRODUCT) ARE IN THE TABLE OF TEST CASES, IN INTERMEDIATE FORMAT. THE TWO SOURCES ARE MULTIPLIED TOGETHER (IN SIGNED PACKED FORMAT) AND THE EXPECTED PRODUCT IS COMPARED WITH THE ACTUAL RESULT. ALSO, THE EXPECTED VALUES OF R0-R5 AND THE CONDITION CODES ARE COMPARED WITH THE ACTUAL RESULTS. ANY DISCREPANCIES ARE FLAGGED BY ERROR CALLS. UNSIGNED PACKED FORMAT IS TESTED IN THE INLINE CASE ONLY.

DIVP - SAME AS THE MULP TEST, EXCEPT THAT A DIVP IS EXECUTED AND THE TABLE OF TEST CASES CONTAINS QUOTIENT RESULTS FOR THE LAST TWO TABLE ENTRIES (DESTINATION POINTER AND CC.EXP).

7.0 REPAIR HISTORY

DATE	EDIT TRAIL	PROG	REASON
12/09/83	PA001	P. ANASTAS	TO MAKE DIAGNOSTIC COMPATIBLE WITH USER FRIENDLY STRATEGY.

```

760
761      ;*COPYRIGHT (C) 1981
762      ;*DIGITAL EQUIPMENT CORP.
763      ;*MAYNARD, MASS. 01754
764      ;*
765      ;*
766      ;*THIS PROGRAM WAS ASSEMBLED USING SPMACJ
767      ;*
768
769      .SBTTL  BASIC DEFINITIONS
770
771      ;*INITIAL ADDRESS OF THE STACK POINTER *** 776 ***
772
773      000776      STACK= 776
774      ;ERROR= EMT      ;;BASIC DEFINITION OF ERROR CALL
775      000004      SCOPE =IOT      ;;BASIC DEFINITION OF SCOPE CALL
776
777      ;*MISCELLANEOUS DEFINITIONS
778
779      000011      HT= 11      ;;CODE FOR HORIZONTAL TAB
780      000012      LF= 12      ;;CODE FOR LINE FEED
781      000015      CR= 15      ;;CODE FOR CARRIAGE RETURN
782      000200      CRLF= 200      ;;CODE FOR CARRIAGE RETURN-LINE FEED
783      177776      PS= 177776      ;;PROCESSOR STATUS WORD
784      177776      PSW= PS
785      177774      STKLMT= 177774      ;;STACK LIMIT REGISTER
786      177546      LKS= 177546      ;;LINE CLOCK STATUS REGISTER
787      000100      BUFLen= 100      ;LENGTH OF BUFF1 AND BUFF2
788
789      ;*CONSOLE REGISTER DEFINITIONS
790
791      177560      TKS= 177560
792      177562      TKB= 177562
793      177564      TPS= 177564
794      177566      TPB= 177566
795
796      ;*MEMORY MANAGEMENT REGISTERS
797
798      172300      KIPDR0= 172300
799      172302      KIPDR1= 172302
800      172304      KIPDR2= 172304
801      172306      KIPDR3= 172306
802      172310      KIPDR4= 172310
803      172312      KIPDR5= 172312
804      172314      KIPDR6= 172314
805      172316      KIPDR7= 172316
806
807      172340      KIPAR0= 172340
808      172342      KIPAR1= 172342
809      172344      KIPAR2= 172344
810      172346      KIPAR3= 172346
811      172350      KIPAR4= 172350
812      172352      KIPAR5= 172352
813      172354      KIPAR6= 172354
814      172356      KIPAR7= 172356
815
816      177572      SR1= 177572

```

## BASIC DEFINITIONS

```

817
818
819
820          000000      R0=    %0          ;;GENERAL REGISTER
821          000001      R1=    %1          ;;GENERAL REGISTER
822          000002      R2=    %2          ;;GENERAL REGISTER
823          000003      R3=    %3          ;;GENERAL REGISTER
824          000004      R4=    %4          ;;GENERAL REGISTER
825          000005      R5=    %5          ;;GENERAL REGISTER
826          000006      R6=    %6          ;;GENERAL REGISTER
827          000007      R7=    %7          ;;GENERAL REGISTER
828          000006      SP=    %6          ;;STACK POINTER
829          000007      PC=    %7          ;;PROGRAM COUNTER
830
831          ;;*PRIORITY LEVEL DEFINITIONS
832
833          000000      PR0=    0          ;;PRIORITY LEVEL 0
834          000040      PR1=    40         ;;PRIORITY LEVEL 1
835          000100      PR2=    100        ;;PRIORITY LEVEL 2
836          000140      PR3=    140        ;;PRIORITY LEVEL 3
837          000200      PR4=    200        ;;PRIORITY LEVEL 4
838          000240      PR5=    240        ;;PRIORITY LEVEL 5
839          000300      PR6=    300        ;;PRIORITY LEVEL 6
840          000340      PR7=    340        ;;PRIORITY LEVEL 7
841
842          ;;*CONDITION CODE EQUATES
843
844          000010      NBIT=   10
845          000004      ZBIT=    4
846          000002      VBIT=    2
847          000001      CBIT=    1
848
849          ;;*"SWITCH REGISTER" SWITCH DEFINITIONS
850
851          100000      SW15=  100000
852          040000      SW14=   40000
853          020000      SW13=  20000
854          010000      SW12=  10000
855          004000      SW11=   4000
856          002000      SW10=   2000
857          001000      SW09=   1000
858          000400      SW08=   400
859          000200      SW07=   200
860          000100      SW06=   100
861          000040      SW05=   40
862          000020      SW04=   20
863          000010      SW03=   10
864          000004      SW02=    4
865          000002      SW01=    2
866          000001      SW00=    1
867          001000      SW9=    SW09
868          000400      SW8=    SW08
869          000200      SW7=    SW07
870          000100      SW6=    SW06
871          000040      SW5=    SW05
872          000020      SW4=    SW04
873          000010      SW3=    SW03

```

BASIC DEFINITIONS

```

874      000004      SW2=   SW02
875      000002      SW1=   SW01
876      000001      SW0=   SW00
877
878      ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
879
880      100000      BIT15=  100000
881      040000      BIT14=  40000
882      020000      BIT13=  20000
883      010000      BIT12=  10000
884      004000      BIT11=  4000
885      002000      BIT10=  2000
886      001000      BIT09=  1000
887      000400      BIT08=  400
888      000200      BIT07=  200
889      000100      BIT06=  100
890      000040      BIT05=  40
891      000020      BIT04=  20
892      000010      BIT03=  10
893      000004      BIT02=  4
894      000002      BIT01=  2
895      000001      BIT00=  1
896      001000      BIT9=   BIT09
897      000400      BIT8=   BIT08
898      000200      BIT7=   BIT07
899      000100      BIT6=   BIT06
900      000040      BIT5=   BIT05
901      000020      BIT4=   BIT04
902      000010      BIT3=   BIT03
903      000004      BIT2=   BIT02
904      000002      BIT1=   BIT01
905      000001      BIT0=   BIT00
906
907      ;*BASIC "CPU" TRAP VECTOR ADDRESSES
908
909      000004      ERRVEC= 4           ;;TIME OUT AND OTHER ERRORS
910      000010      RESVEC= 10          ;;RESERVED AND ILLEGAL INSTRUCTIONS
911      000014      TBITVEC=14         ;; "T" BIT
912      000014      TRTVEC= 14          ;;TRACE TRAP
913      000014      BPTVEC= 14          ;;BREAKPOINT TRAP (BPT)
914      000020      IOTVEC= 20          ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
915      000024      PWRVEC= 24          ;;POWER FAIL
916      000030      EMTVEC= 30          ;;EMULATOR TRAP (EMT)
917      000034      TRAPVEC=34         ;; "TRAP" TRAP
918      000052      UFDMODE=52         ;;CHECK FOR QUIET MONITOR           ;PA001
919      000001      UFDSET=1           ;;TELL SETUP MACRO THIS IS A UFD DIAGNOSTIC ;PA001
920      000060      TKVEC= 60           ;;TTY KEYBOARD VECTOR
921      000064      TPVEC= 64           ;;TTY PRINTER VECTOR
922      000100      LTCVEC= 100         ;;LTC VECTOR
923      000250      MMVEC= 250          ;;MEMORY MANAGEMENT VECTOR
924      000000      ASWREG =0           ;;INITIAL APT SOFTWARE SWR VALUE
925
926
927      ;*EQUATES FOR ROUTINE CALLS THAT USE TRAP INSTRUCTION
928
929      104401      TYPE= 104401         ;;TTY TYPEOUT ROUTINE
930      104402      TYPOC= 104402       ;;TYPE OCTAL NUMBER (WITH LEADING ZEROES) ROUTINE

```

BASIC DEFINITIONS

```

931      104403      TYPOS= 104403      ;TYPE OCTAL NUMBER (NO LEADING ZEROES) ROUTINE
932      104404      TYPON= 104404      ;TYPE OCTAL NUMBER (AS PER LAST CALL) ROUTINE
933      104405      GTSWR= 104405      ;GET SOFT-SWR SETTING ROUTINE
934      104406      CKSWR= 104406      ;TEST FOR CHANGE IN SOFT-SWR ROUTINE
935      100407      RDCHR= 100407      ;TTY TYPEIN CHARACTER ROUTINE
936      104410      RDLIN= 104410      ;TTY TYPEIN STRING ROUTINE
937
938      ;*****
939      ;      *** COMMERCIAL INSTRUCTION SET OPCODE EQUATES ***
940      ;
941
942      000007      MFPT      =000007      ;MOVE FROM PROCESSOR TYPE
943
944      ;      COMMERCIAL LOAD DESCRIPTORS
945
946      076020      L2D0      =76020      ;LOAD 2 DESCRIPTORS @ (R0).
947      076021      L2D1      =76021      ;LOAD 2 DESCRIPTORS @ (R1).
948      076022      L2D2      =76022      ;LOAD 2 DESCRIPTORS @ (R2).
949      076023      L2D3      =76023      ;LOAD 2 DESCRIPTORS @ (R3).
950      076024      L2D4      =76024      ;LOAD 2 DESCRIPTORS @ (R4).
951      076025      L2D5      =76025      ;LOAD 2 DESCRIPTORS @ (R5).
952      076026      L2D6      =76026      ;LOAD 2 DESCRIPTORS @ (R6).
953      076027      L2D7      =76027      ;LOAD 2 DESCRIPTORS @ (R7).
954
955      076060      L3D0      =76060      ;LOAD 3 DESCRIPTORS @ (R0).
956      076061      L3D1      =76061      ;LOAD 3 DESCRIPTORS @ (R1).
957      076062      L3D2      =76062      ;LOAD 3 DESCRIPTORS @ (R2).
958      076063      L3D3      =76063      ;LOAD 3 DESCRIPTORS @ (R3).
959      076064      L3D4      =76064      ;LOAD 3 DESCRIPTORS @ (R4).
960      076065      L3D5      =76065      ;LOAD 3 DESCRIPTORS @ (R5).
961      076066      L3D6      =76066      ;LOAD 3 DESCRIPTORS @ (R6).
962      076067      L3D7      =76067      ;LOAD 3 DESCRIPTORS @ (R7).
963
964
965      ;      CHARACTER STRING INSTRUCTIONS
966
967      076030      MOVCI     =76030      ;MOVE CHARACTER
968      076031      MOVRCI    =76031      ;MOVE REVERSE CHARACTER
969      076032      MOVTCI    =76032      ;MOVE TRANSLATED CHARACTER
970      076040      LOCCI     =76040      ;LOCATE CHARACTER
971      076041      SKPCI     =76041      ;SKIP CHARACTER
972      076042      SCANCI    =76042      ;SCAN CHARACTER
973      076043      SPANCI    =76043      ;SPAN CHARACTER
974      076044      CMPCI     =76044      ;COMPARE CHARACTER
975      076045      MATCI     =76045      ;MATCH CHARACTER
976
977      076130      MOVCI     =76130      ;IN-LINE VERSION OF MOVE CHARACTER
978      076131      MOVRCI    =76131      ;IN-LINE VERSION
979      076132      MOVTCI    =76132
980      076140      LOCCI     =76140
981      076141      SKPCI     =76141
982      076142      SCANCI    =76142
983      076143      SPANCI    =76143
984      076144      CMPCI     =76144
985      076145      MATCI     =76145
986
987

```

BASIC DEFINITIONS

```

988      ;          NUMERIC STRING INSTRUCTIONS
989
990      076050     ADDN      =76050      ;ADD NUMERIC
991      076051     SUBN      =76051      ;SUBTRACT NUMERIC
992      076052     CMPN      =76052      ;COMPARE NUMERIC
993      076053     CVTNL     =76053      ;CONVERT NUMERIC TO LONG
994      076054     CVTPN     =76054      ;CONVERT PACKED TO NUMERIC
995      076055     CVTNP     =76055      ;CONVERT NUMERIC TO PACKED
996      076056     ASHN      =76056      ;ARITHMETIC SHIFT NUMERIC
997      076057     CVTLN     =76057      ;CONVERT LONG TO NUMERIC
998
999      076150     ADDNI     =76150      ;IN-LINE VERSION OF ADD NUMERIC
1000     076151     SUBNI     =76151      ;IN-LINE VERSION
1001     076152     CMPNI     =76152      ;
1002     076153     CVTNLI    =76153      ;
1003     076154     CVTPNI    =76154      ;
1004     076155     CVTNPI    =76155      ;
1005     076156     ASHNI     =76156      ;
1006     076157     CVTLNI    =76157      ;
1007
1008
1009     ;          PACKED STRING INSTRUCTIONS
1010
1011     076070     ADDP      =76070      ;ADD PACKED
1012     076071     SUBP      =76071      ;SUBTRACT PACKED
1013     076072     CMPP      =76072      ;COMPARE PACKED
1014     076073     CVTPL     =76073      ;CONVERT PACKED TO LONG
1015     076074     MULP      =76074      ;MULTIPLY PACKED
1016     076075     DIVP      =76075      ;DIVIDE PACKED
1017     076076     ASHP      =76076      ;ARITHMETIC SHIFT PACKED
1018     076077     CVTLP     =76077      ;CONVERT LONG TO PACKED
1019
1020     076170     ADDPI     =76170      ;IN-LINE VERSION OF ADD PACKED
1021     076171     SUBPI     =76171      ;IN-LINE VERSION
1022     076172     CMPPI     =76172      ;
1023     076173     CVTPLI    =76173      ;
1024     076174     MULPI     =76174      ;
1025     076175     DIVPI     =76175      ;
1026     076176     ASHPI     =76176      ;
1027     076177     CVTLPI    =76177      ;
1028
1029
1030     ;          CIS CLASS ILLEGAL OP CODE
1031
1032     076033     ILLOP     =76033      ;ILLEGAL OP CODE
1033
1034
1035
1036     ;*****
1037
1038
1039     .SBTTL TRAP CATCHER
1040
1041     000000     .ASECT
1042     000000     .=0
1043
1058     ;*ALL UNUSED LOCATIONS FROM 4 776 CONTAIN A '.+2.HALT'

```



TRAP CATCHER

```

1059      ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
1060      ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
1061
1062      000174      . =174
1063
1064 000174 000000  DISPREG: .WORD 0      ;;SOFTWARE DISPLAY REGISTER
1065 000176 000000  SWREG:   .WORD 0      ;;SOFTWARE SWITCH REGISTER
1066
1067
1068      000100      . =100      ;HANDLE EVENT LINE INTERRUPT
1069
1070 000100 000102  .WORD 102
1071 000102 000002  .WORD 2
1072
1073      000200      . =200
1074
1075 000200 000167 000774  JMP     START      ;STARTING LOCATION FOR PROGRAM
1076
1077      000300      . =300
1078
1079      .SBTTL  ACT11 HOOKS
1080
1081      ;;*****
1082      ;HOOKS REQUIRED BY ACT11
1083      000300      $SVPC=.      ;SAVE PC
1084      000046      . =46
1085 000046 056232  $ENDAD      ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
1086      000052      . =52
1087 000052 000000  .WORD 0      ;;2)SET LOC.52 TO ZERO
1088      000300      . =$SVPC      ;; RESTORE PC
1089
1090      .SBTTL  APT PARAMETER BLOCK
1091
1092      ;;*****
1093      ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
1094      ;;*****
1095      000300      . $X=.      ;;SAVE CURRENT LOCATION
1096      000024      . =24      ;;SET POWER FAIL TO POINT TO START OF PROGRAM
1097 000024 000200  200      ;;FOR APT START UP
1098      000044      . =44      ;;POINT TO APT INDIRECT ADDRESS PNTR.
1099 000044 000300  $APTHDR ;;POINT TO APT HEADER BLOCK
1100      000300      . =.$X      ;;RESET LOCATION COUNTER
1101
1102      ;;*****
1103      ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
1104      ;INTERFACE SPEC.
1105
1106 000300  $APTHD:
1107 000300 000000  $HIBTS: .WORD 0      ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
1108 000302 000632  $MADR:  .WORD $MAIL  ;;ADDRESS OF APT MAILBOX (BITS 0-15)
1109 000304 000300  $TSTM:  .WORD 300    ;;RUN TIM OF LONGEST TEST
1110 000306 000300  $PASTM: .WORD 300    ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
1111 000310 000000  $UNITM: .WORD      ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
1112 000312 000015  .WORD $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
1113      .SBTTL  COMMON TAGS
1114
1115      ;;*****

```

COMMON TAGS

```

1116          ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
1117          ;*USED IN THE PROGRAM.
1118
1119          000400          .-400
1120
1121 000400    $CMTAG:          ;; START OF COMMON TAGS
1122 000400    .WORD          0
1123 000402    $TSTNM: .WORD  0          ;; CONTAINS THE TEST NUMBER
1124 000404    $TIMES: .WORD  0          ;; CONTAINS NUMBER OF ITERATIONS DESIRED
1125 000406    $ICNT:  .WORD  0          ;; CONTAINS SUBTEST ITERATION COUNT
1126 000410    $LPADR: .WORD  0          ;; CONTAINS SCOPE LOOP ADDRESS
1127 000412    $LPERR: .WORD  0          ;; CONTAINS SCOPE RETURN FOR ERRORS
1128 000414    $ERTTL: .WORD  0          ;; CONTAINS TOTAL ERRORS DETECTED
1129 000416    $ERMAX: .BYTE  1          ;; CONTAINS MAX. ERRORS PER TEST
1130          .EVEN
1131 000420    $ERRPC: .WORD  0          ;; CONTAINS PC OF LAST ERROR INSTRUCTION
1132 000422    $AUTOB: .BYTE  0          ;; AUTOMATIC MODE INDICATOR
1133 000423    $INTAG: .BYTE  0          ;; INTERRUPT MODE INDICATOR
1134 000424    .WORD          0
1135 000426    $SWR:  .WORD  SWREG        ;; ADDRESS OF SWITCH REGISTER
1136 000430    $DISPLAY: .WORD  DISPREG   ;; ADDRESS OF DISPLAY REGISTER
1137 000432    $NULL:  .BYTE  0          ;; CONTAINS NULL CHARACTER FOR FILLS
1138 000433    $FILLS: .BYTE  2          ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
1139 000434    $FILLC: .BYTE  12         ;; INSERT FILL CHARS. AFTER A "LINE FEED"
1140 000435    $TPFLG: .BYTE  0          ;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
1141 000436    $BELL:  .ASCIZ <207><377><377> ;; CODE FOR BELL
1142 000441    000
1142 000442    $QUES:  .ASCII  /?/?      ;; QUESTION MARK
1143 000443    $CRLF:  .ASCII  <15>      ;; CARRIAGE RETURN
1144 000444    $LF:   .ASCIZ  <12>      ;; LINE FEED
1145
1146
1147 000446    000000    MMCYL: .WORD  0          ;MEMORY MANAGEMENT CYCLE FLAG: 0=KTOFF, 1=KTON
1148 000450    000000    SIGNO: .WORD  0          ;SIGN FLAG: 0 FOR +, 1 FOR -
1149 000452    000000    SIGN1: .WORD  0          ;
1150 000454    000000    SIGN2: .WORD  0          ;
1151 000456    000000    ERRFLG: .WORD  0          ;ERROR FLAG
1152 000460    000000    INTFLG: .WORD  0          ;INTERRUPT FLAG
1153 000462    000000    INTPAS: .WORD  0          ;FLAGS FIRST PASS OF INTERRUPT TEST
1154 000464    000000    DATTYP: .WORD  0          ;DECIMAL STRING DATA TYPE FLAG
1155 000466    000000    XXDP:  .WORD  0          ;IF NOT 0 THEN WE ARE IN XXDP CHAIN MODE
1156 000470    000000    COUNT: .WORD  0          ;TEMPORARY COUNTER
1157 000472    000000    DELAY: .WORD  0          ;TIME-DELAY COUNT STORED HERE
1158 000474    000000    ERRTMP: .WORD  0
1159 000476    000000    TEMPO: .WORD  0          ;TEMPORARY VARIABLE
1160 000500    000000    TEMP1: .WORD  0          ;TEMPORARY VARIABLE
1161 000502    000000    TEMP2: .WORD  0          ;TEMPORARY VARIABLE
1162 000504    000000    TEMP3: .WORD  0          ;TEMPORARY VARIABLE
1163 000506    000000    TEMP4: .WORD  0          ;TEMPORARY VARIABLE
1164 000510    000000    TEMP5: .WORD  0          ;TEMPORARY VARIABLE
1165 000512    000000    OFFSET: .WORD  0
1166 000514    000000    DSLEN1: .WORD  0          ;FIRST DECIMAL STRING LENGTH
1167 000516    000000    DSLEN2: .WORD  0          ;SECOND DECIMAL STRING LENGTH
1168 000520    000000    CC:     .WORD  0          ;CONDITION CODES MAY BE SAVED HERE
1169 000522    000000    CC.EXP: .WORD  0          ;EXPECTED CONDITION CODES
1170 000524    000000    RO.EXP: .WORD  0          ;EXPECTED VALUE OF R0
1171 000526    000000    R1.EXP: .WORD  0          ;

```

COMMON TAGS

```

1172 000530 000000 R2.EXP: .WORD 0 ; R2
1173 000532 000000 R3.EXP: .WORD 0 ; R3
1174 000534 000000 R4.EXP: .WORD 0 ; R4
1175 000536 000000 R5.EXP: .WORD 0 ; R5
1176 000540 000000 BF.EXP: .WORD 0 ; EXPECTED BUFFER DATA
1177 000542 000000 BFADRS: .WORD 0 ; BUFFER POINTER
1178 000544 000000 TSTCAS: .WORD 0 ; POINTER TO CURRENT TEST CASE IN TEST TABLE
1179 000546 000000 CTL: .WORD 0 ; CONTROL CHIP NUMBER
1180 000550 000000 MSGADR: .WORD 0 ; ADDRESS OF THE ASCII TO BE PRINTED
1181 000552 000000 ERR: .WORD 0 ; ERROR TYPE NUMBER
1182 000554 000 DSTMP0: .BYTE 0 ; DECIMAL STRING TEMPORARY
1183 000555 000 DSTMP1: .BYTE 0 ; DECIMAL STRING TEMPORARY
1184 000556 000 DSTMP2: .BYTE 0 ; DECIMAL STRING TEMPORARY
1185 000557 000 DSTMP3: .BYTE 0 ; DECIMAL STRING TEMPORARY
1186 000560 000 DSTMP4: .BYTE 0 ; DECIMAL STRING TEMPORARY
1187 000561 000 DSTMP5: .BYTE 0 ; DECIMAL STRING TEMPORARY
1188 000562 000000 DATA0: .WORD 0 ; TEST DATA GETS LOADED HERE
1189 000564 000000 DATA01: .WORD 0 ;
1190 000566 000000 DATA02: .WORD 0 ;
1191 000570 000000 DATA03: .WORD 0 ;
1192 000572 000000 DATA04: .WORD 0 ;
1193 000574 000000 DATA05: .WORD 0 ;
1194 000576 000000 DATA06: .WORD 0 ;
1195
1196 000600 000000 SAVR0: .WORD 0 ; GENERAL PURPOSE REGISTERS MAY BE SAVED HERE
1197 000602 000000 SAVR1: .WORD 0 ;
1198 000604 000000 SAVR2: .WORD 0 ;
1199 000606 000000 SAVR3: .WORD 0 ;
1200 000610 000000 SAVR4: .WORD 0 ;
1201 000612 000000 SAVR5: .WORD 0 ;
1202 000614 000000 SAVPSW: .WORD 0 ; PSW MAY BE SAVED HERE
1203 000616 000000 DESCRO: .WORD 0 ; TEMPS FOR DS DESCRIPTORS
1204 000620 000000 DESCR1: .WORD 0
1205 000622 000000 DESCR2: .WORD 0
1206 000624 000000 DESCR3: .WORD 0
1207 000626 000000 DESCR4: .WORD 0
1208 000630 000000 DESCR5: .WORD 0
1209
1210
1211 ;*****
1212 .SBTTL APT MAILBOX-ETABLE
1213 ;*****
1214 ;
1215 ;
1216 ;
1217
1218 .EVEN
1219 000632 $MAIL: ; APT MAILBOX
1220 000632 000000 $MSGTY: .WORD 000000 ; MESSAGE TYPE CODE
1221 000634 000000 $FATAL: .WORD 000000 ; FATAL ERROR NUMBER
1222 000636 000000 $TESTN: .WORD 000000 ; TEST NUMBER
1223 000640 000000 $PASS: .WORD 000000 ; PASS COUNT
1224 000642 000000 $DEVCT: .WORD 000000 ; DEVICE COUNT
1225 000644 000000 $UNIT: .WORD 000000 ; I/O UNIT NUMBER
1226 000646 000000 $MSGAD: .WORD 000000 ; MESSAGE ADDRESS
1227 000650 000000 $MSGLG: .WORD 000000 ; MESSAGE LENGTH
1228 000652 $ETABLE: ; APT ENVIRONMENT TABLE

```

APT MAILBOX-ETABLE

```

1229 000652      000      $ENV:      .BYTE      000      ;;ENVIRONMENT BYTE
1230 000653      000      $ENVM:     .BYTE      000      ;;ENVIRONMENT MODE BITS
1231 000654     000000    $SWREG:    .WORD      000000   ;;APT SWITCH REGISTER
1232 000656     000000    $USWR:     .WORD      000000   ;;USER SWITCHES
1233 000660     000000    $CPUOP:    .WORD      000000   ;;CPU TYPE,OPTIONS
1234              ;*              BITS 15-11=CPU TYPE
1235              ;*              11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
1236              ;*              11/70=06,PDQ=07,Q=10
1237              ;*              BIT 10=REAL TIME CLOCK
1238              ;*              BIT 9=FLOATING POINT PROCESSOR
1239              ;*              BIT 8=MEMORY MANAGEMENT
1240 000662      000      $MAMS1:    .BYTE      000      ;;HIGH ADDRESS,M.S. BYTE
1241 000663      000      $MTYP1:    .BYTE      000      ;;MEM. TYPE, BLK#1
1242              ;*              MEM. TYPE BYTE -- (HIGH BYTE)
1243              ;*              900 NSEC CORE=001
1244              ;*              300 NSEC BIPOLAR=002
1245              ;*              500 NSEC MOS=003
1246 000664      $ETEND:
1247              .EVEN
1248
1249
1250
1251              ;;*****
1252              .SBTTL  START UP AND INITIALIZATION ROUTINES
1253              ;;*****
1254
1255              ;          .PSECT  CODE,REL,CON,GBL,I,RW
1256              .-1200
1257 001200      001200    START::
1258
1259              .SBTTL  INITIALIZE THE COMMON TAGS
1260
1261              ;;CLEAR THE COMMON TAGS ($CHTAG) AREA
1262              .LIST  MEB
1263 001200      001200    .SETUP              ;PA001
1264 001200      005767    000062    EMTSAV:  TST      SAV30              ;; FIRST TIME THROUGH ?
1265 001204      001034          BNE      VMKOR              ;; BRANCH IF BEEN HERE ALREADY
1266 001206      032737    000040    000052    BIT      @BIT5,@#52          ;; ARE WE IN UFD MODE ?
1267 001214      001430          BEQ      VMKOR              ;; LEAVE IF NOT
1268 001216      012767    177777    000046    MOV      @-1,UFDPLG          ;; SET UFD FLAG
1269 001224      032737    000100    000052    BIT      @BIT6,@#52          ;; ARE WE IN QUIET MODE ?
1270 001232      001403          BEQ      1#              ;; BR IF NOT
1271 001234      012767    177777    000032    MOV      @-1,UQUIET          ;; SET QUIET MODE
1272 001242      104042          1#:      EMT      42              ;; GET ADDRESS OF XXDP DCA TABLE
1273 001244      005060    000042          CLR      42(R0)           ;; CLR XXDP- "DRSERR"
1274 001250      016767    176554    000010    MOV      30,SAV30           ;; SAVE EMULATOR ADDRESS
1275 001256      016767    176550    000004    MOV      32,SAV32           ;; SAVE EMULATOR PRIORITY LEVEL
1276 001264      000404          BR      VMKOR              ;; GET AROUND TAG AREA
1277 001266      000000          SAV30:  .WORD      0              ;; PUT EMULATOR INFO HERE
1278 001270      000000          SAV32:  .WORD      0              ;; PUT PRIORITY LOCATION      HERE
1279 001272      000000          UFDPLG: .WORD      0              ;; USER FRIENDLY MODE FLAG
1280 001274      000000          UQUIET: .WORD      0              ;; UFD QUIET MODE FLAG
1281
1282 1264
1283 1265 001276    012767    177777    177770    MOV      @-1,UQUIET          ;;TO RETURN TO UFD MODE ON HALT      ;PA001
1284 1266 001304    012706    000400          MOV      @CHTAG,R6          ;;FIRST LOCATION TO BE CLEARED
1285 1267 001310    005026          CLR      (R6)              ;;CLEAR MEMORY LOCATION
1286 1268 001312    022706    000426          CMP      @SWR,R6            ;;DONE?

```

INITIALIZE THE COMMON TAGS

```

1269 001316 001374          BNE      -6          ;;LOOP BACK IF NO
1270 001320 012706 001100    MOV      @1100,SP      ;;SETUP THE STACK POINTER
1271
1272          ;;INITIALIZE A FEW VECTORS
1273
1274 001324 012737 056354 000004    MOV      @TIMOUT,@ERRVEC      ;TIME-OUT VECTOR
1275 001332 012737 000340 000006    MOV      @340,@ERRVEC+2      ;LEVEL 7
1276 001340 012737 056406 000010    MOV      @ILLRES,@RESVEC     ;ILLEGAL/RESERVED INSTR. VECTOR
1277 001346 012737 000340 000012    MOV      @340,@RESVEC+2      ;LEVEL 7
1278 001354 012737 057376 000020    MOV      @$SCOPE,@IOTVEC     ;;IOT VECTOR FOR SCOPE ROUTINE
1279 001362 012737 000340 000022    MOV      @340,@IOTVEC+2      ;;LEVEL 7
1280 001370 012737 062320 000034    MOV      @$TRAP,@TRAPVEC     ;;TRAP VECTOR FOR TRAP CALLS
1281 001376 012737 000340 000036    MOV      @340,@TRAPVEC+2     ;LEVEL 7
1282 001404 012737 057516 000024    MOV      @$PWRDN,@PWRVEC     ;;POWER FAILURE VECTOR
1283 001412 012737 000340 000024    MOV      @340,@PWRVEC+2     ;;LEVEL 7
1284 001420 016767 054662 054656    MOV      $ENDCT,$EOPCT      ;;SETUP END-OF-PROGRAM COUNTER
1285 001426 012767 001426 176754    MOV      @,$LPADR           ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
1286 001434 012737 000102 000100    MOV      @102,@LTCVEC       ;GO TO 102
1287 001442 012737 000002 000102    MOV      @2,@102           ;RTI
1288 001450 012737 000252 000250    MOV      @252,@MMVEC        ;SET UP KT VECTOR
1289 001456 012737 177777 000446    MOV      @-1,@MMCYL         ;INITIALIZE MEMORY MANAGEMENT CYCLE FLAG
1290
1291          ;;SET UP SOFTWARE SWITCH REGISTER
1292
1293 001464 012767 000176 176734 65$:  MOV      @SWREG,SWR          ;;POINT TO SOFTWARE SWR
1294 001472 012767 000174 176730    MOV      @DISPREG,DISPLAY
1295
1296 001500 005067 177134          CLR      $PASS              ;;CLEAR PASS COUNT
1297 001504 132767 000200 177141    BITB    @APTSIZE,$ENVM      ;;TEST USER SIZE UNDER APT
1298 001512 001403          BEQ     67$                ;;YES,USE NON-APT SWITCH
1299 001514 012767 000654 176704    MOV      @$SWREG,SWR        ;;NO,USE APT SWITCH REGISTER
1300 001522
1301
1302          ;;DETERMINE OPERATING ENVIRONMENT
1303
1304
1305 001522 032767 000040 176322    BIT     @BITS,UFDMODE       ;ARE WE RUNNING UNDER THE UFD?          ;PA001
1306 001530 001403          BEQ     6$                 ;NO, NO NEED TO CLEAR DRSEERR          ;PA001
1307 001532 104042          EMT     42                 ;YES, GET ADDRESS OF DCA INTO R0       ;PA001
1308 001534 005060 000042          CLR     42(R0)             ;CLEAR DRSEERR                          ;PA001
1309 001540 005737 000042          6$:   TST     @42            ;ARE WE IN ACT/XXDP AUTO MODE?         ;PA001
1310 001544 001410          BEQ     1$                 ;BRANCH IF NO
1311 001546 023737 000042 000046    CMP     @42,@46            ;IS IT ACT AUTO MODE?
1312 001554 001410          BEQ     2$                 ;BRANCH IF YES
1313 001556 012767 000001 176702    MOV     @1,XXDP            ;SET XXDP CHAIN MODE INDICATOR         ;PA001
1314 001564 000404          BR     2$                  ;
1315 001566 126727 177060 000001 1$:   CMPB   $ENV,@1            ;ARE WE IN APT AUTO MODE?
1316 001574 001003          BNE     3$                 ;BRANCH IF NO
1317 001576 112767 000001 176616 2$:   MOVB   @1,$AUTOB         ;SET AUTO MODE INDICATOR
1318
1319          ;PRINT TITLE IF NOT IN ACT OR APT AUTO MODE
1320
1321 001604 005227 177777          3$:   INC     @-1              ;FIRST TIME?
1322 001610 001007          BNE     5$                 ;SKIP TITLE IF NOT
1323 001612 005767 176604          TST     $AUTOB            ;ARE WE IN AUTO MODE?
1324 001616 001004          BNF     5$                 ;BRANCH TO TITLE TYPEOUT IF NOT
1325 001620 012767 062376 176722 4$:   MOV     @TITLE,MSGADR

```

INITIALIZE THE COMMON TAGS

```

1326 001626 104401          TYPE          ;PRINT OUT THE TITLE
1327
1328          ;GET THE VALUE IN THE SOFTWARE SWITCH REGISTER
1329
1330 001630 005767 176566  S4:   TST     $AUTOB      ;ARE WE IN AN AUTOMATIC MODE?
1331 001634 001001          BNE     BEGIN      ;BRANCH IF YES
1332 001636 104405          GTSWR          ;ASK FOR SWR INPUT FROM THE CONSOLE TERMINAL
1333
1334 001640          BEGIN:
1335 001640          CALL MMSETUP          ;GO CYCLE MEMORY MANAGEMENT
1336 001640 004767 055156          JSR     PC,MMSETUP

```

INITIALIZE THE COMMON TAGS

```

1338 ;*****
1339 ;*****
1340 ;**
1341 ;           C I S 4   M O D U L E
1342 ;
1343 ;   THIS MODULE TESTS THE LOAD DESCRIPTOR INSTRUCTIONS AND THE CHARACTER
1344 ;   INSTRUCTIONS:  L2DR, L3DR, MOV, MOVRC, MOVTC, LOCC, SKPC, SCANC,
1345 ;   SPANC, CMPC, AND MATC.
1346 ;--
1347 ;*****
1348 ;*****
1349
1350
1351
1352 .SBTTL  CIS4 INITIALIZATION
1353 ;*****
1354 ;
1355 ;   CIS4 INITIALIZATION
1356 ;
1357 ;*****
1358
1359 ;IF SWR<5:0> ARE CLEARED OR IF SWR<0> IS SET THEN EXECUTE
1360 ; THE CIS4 MODULE; OTHERWISE, SKIP TO THE NEXT CIS MODULE.
1361
1362 001644      INLINE  <SCOPE>
1363 001644 000004      SCOPE
1364 001646      INLINE <CKSWR>
1365 001646 104406      CKSWR
1366 001650      LET TEMPO := @SWR CLR.BY #177700
1367 001650 017767 176552 176620      MOV  @SWR,TEMPO
1368 001656 042767 177700 176612      BIC  #177700,TEMPO
1369 001664      IF TEMPO NE #0 THEN
1370 001664 005767 176606      TST  TEMPO
1371 001670 001406      BEQ  $50000
1372 001672      IF #BIT0 NOTSETIN TEMPO THEN
1373 001672 032767 000001 176576      BIT  #BIT0,TEMPO
1374 001700 001002      BNE  $50001
1375 001702      INLINE  <JMP  CIS4END>
1376 001702 000167 014714      JMP  CIS4END
1377 001706      ENDIF
1378 001706      $50001:
1379 001706      ENDIF
1380 001706      $50000:
1381
1382 001706      LET CTL := #4 ;CONTROL CHIP NUMBER
1383 001706 012767 000004 176632      MOV  #4,CTL
1384 001714      LET ERRFLG := #0 ;INITIALIZE ERROR FLAG
1385 001714 005067 176536      CLR  ERRFLG
1386 001720      LET PSW := #340
1387 001720 012767 000340 176050      MOV  #340,PSW
1388

```

L2DR

```

1377 .SBTTL L2DR
1378 ;*****
1379 ;
1380 ; L2DR TEST
1381 ; THIS TEST USES L2D0, L2D4, AND L2D7 TO TEST L2DR.
1382 ;
1383 ;*****
1384
1385 ROUTINE XL2DR
1386 001726 001726 XL2DR:
1387 001726 000240      NOP
1388 001730 000004      SCOPE
1389 001732 012767 000020 176444 LET $TIMES := #20 ;ITERATION COUNT
1390 001740 012767 000002 176522 LET COUNT := #2
1391 001746 001746 REPEAT
1392 ;*
1393 ; SET UP THE GPR'S AND DO
1394 ; THE CIS INSTRUCTION
1395 ;-
1396
1397 001746 004767 054466 CALL CLRREG ;CLEAR R0 R5
1398 001752 026727 176512 000002 IF COUNT EQ #2 THEN
1399 001760 001005      CMP COUNT,#2
1400 001762 012700 002332      BNE $50005
1401 001766 000257      MOV #ADR.A,R0
1402 001770 076020      CCC
1403 001772 000404      L2D0
1404 001774 012704 002332 ELSE
1405 002000 000277      BR $50006
1406 002002 076024      $50005:
1407 002004 016767 175766 176506 LET R4 := #ADR.A
1408 002012 042767 177760 176500      MOV #ADR.A,R4
1409 002020 012767 000001 176524      SCC
1410 ;*
1411 ; CHECK RESULTS AND REPORT ERRORS
1412

```



L2DR

```

1413
1414
1415 002026          IF COUNT EQ #2 THEN
      002026 026727 176436 000002          CMP      COUNT,#2
      002034 001006          BNE      $50007
1416 002036          IF CC NE #0 THEN
      002036 005767 176456          TST      CC
      002042 001402          BEQ      $50010
1417 002044          CALL ERROR
      002044 004767 055630          JSR      PC.ERROR
1418 002050          ENDIF
      002050          $50010:
1419 002050          ELSE
      002050 000406          BR      $50011
      002052          $50007:
1420 002052          IF CC NE #17 THEN
      002052 026727 176442 000017          CMP      CC,#17
      002060 001402          BEQ      $50012
1421 002062          CALL ERROR
      002062 004767 055612          JSR      PC.ERROR
1422 002066          ENDIF
      002066          $50012:
1423 002066          ENDIF
      002066          $50011:
1424 002066          IF R0 NE ALPHA THEN
      002066 020067 000246          CMP      R0,ALPHA
      002072 001402          BEQ      $50013
1425 002074          CALL ERROR
      002074 004767 055600          JSR      PC.ERROR
1426 002100          ENDIF
      002100          $50013:
1427 002100          IF R1 NE ALPHA+2 THEN
      002100 020167 000236          CMP      R1,ALPHA+2
      002104 001402          BEQ      $50014
1428 002106          CALL ERROR
      002106 004767 055566          JSR      PC.ERROR
1429 002112          ENDIF
      002112          $50014:
1430 002112          IF R2 NE BETA THEN
      002112 020267 000226          CMP      R2,BETA
      002116 001402          BEQ      $50015
1431 002120          CALL ERROR
      002120 004767 055554          JSR      PC.ERROR
1432 002124          ENDIF
      002124          $50015:
1433 002124          IF R3 NE BETA+2 THEN
      002124 020367 000216          CMP      R3,BETA+2
      002130 001402          BEQ      $50016
1434 002132          CALL ERROR
      002132 004767 055542          JSR      PC.ERROR
1435 002136          ENDIF
      002136          $50016:
1436 002136          IF R5 NE #0 THEN
      002136 005705          TST      R5
      002140 001402          BEQ      $50017
1437 002142          CALL ERROR
      002142 004767 055532          JSR      PC.ERROR

```

L2DR

1438	002146			ENDIF				
	002146						\$50017:	
1439	002146			IF COUNT EQ #2 THEN				
	002146	026727	176316					CMP
	002154	001005						BNE
1440	002156			IF R4 NE #0 THEN				COUNT, #2
	002156	005704						\$50020
	002160	001402						TST
1441	002162			CALL ERROR				R4
	002162	004767	055512					\$50021
1442	002166			ENDIF				BEQ
	002166							JSR
1443	002166			ELSE			\$50021:	PC, ERROR
	002166	000405						BR
	002170						\$50020:	\$50022
1444	002170			IF R4 NE #ADR.A+4 THEN				
	002170	020427	02336					CMP
	002174	001402						BEQ
1445	002176			CALL ERROR				R4, #ADR.A+4
	002176	004767	055476					\$50023
1446	002202			ENDIF				JSR
	002202							PC, ERROR
1447	002202			ENDIF			\$50023:	
	002202						\$50022:	
1448	002202			LET COUNT := COUNT - #1				
	002202	005367	176262					DEC
1449	002206			UNTIL COUNT EQ #0				COUNT
	002206	005767	176256					TST
	002212	001255						BNE
1450								COUNT
1451								\$50004
1452				;*				
1453				; TEST THE SPECIAL CASE OF USING R7				
1454				;				
1455	002214			LET R4 := #0				
	002214	005004						CLR
1456	002216			LET R5 := #0				R4
	002216	005005						CLR
1457	002220			INLINE <L2D7>				R5
	002220	076027						L2D7
1458	002222			INLINE <.WORD 240>		;NOP		
	002222	000240						.WORD
1459	002224			INLINE <.WORD 402>		;BR TO ERROR CALL		240
	002224	000402						.WORD
1460	002226			INLINE <.WORD 403>		;BR OVER ERROR CALL		402
	002226	000403						.WORD
1461	002230			INLINE <.WORD 240>		;NOP		403
	002230	000240						.WORD
1462	002232			CALL ERROR				240
	002232	004767	055442					JSR
1463	002236			IF R0 NE @#240 THEN				PC, ERROR
	002236	020037	000240					JSR
	002242	001402						CMP
1464	002244			CALL ERROR				R0, @#240
	002244	004767	055430					\$50024
1465	002250			ENDIF				BEQ
	002250							JSR
							\$50024:	PC, ERROR

L2DR

```

1466 002250          IF R1 NE @#242 THEN
      002250 020137 000242
      002254 001402
1467 002256          CALL ERROR
      002256 004767 055416
1468 002262          ENDIF
      002262
1469 002262          IF R2 NE @#402 THEN
      002262 )20237 000402
      002266 )01402
1470 002270          CALL ERROR
      002270 004767 055404
1471 002274          ENDIF
      002274
1472 002274          IF R3 NE @#404 THEN
      002274 020337 000404
      002300 001402
1473 002302          CALL ERROR
      002302 004767 055372
1474 002306          ENDIF
      002306
1475 002306          IF R4 NE #0 THEN
      002306 005704
      002310 001402
1476 002312          CALL ERROR
      002312 004767 055362
1477 002316          ENDIF
      002316
1478 002316          IF R5 NE #0 THEN
      002316 005705
      002320 001402
1479 002322          CALL ERROR
      002322 004767 055352
1480 002326          ENDIF
      002326
1481 002326          INLINE <JMP XL3DR>
      002326 000167 000022
1482
1483
1484
1485
1486
1487 002332 002340
1488 002334 002344
1489 002336 002350
1490 002340 177777
1491 002342 052525
1492 002344 125252
1493 002346 177777
1494 002350 107070
1495 002352 070707
1496

```

```

;+
; TEST TABLE FOR L2DR AND L3DR
;-
ADR.A:  .WORD  ALPHA
        .WORD  BETA
        .WORD  GAMMA
ALPHA:  .WORD  177777      ; ALPHA.LEN
        .WORD  052525      ; ALPHA.ADR
BETA:   .WORD  125252      ; BETA.LEN
        .WORD  177777      ; BETA.ADR
GAMMA:  .WORD  107070      ; GAMMA.LEN
        .WORD  070707      ; GAMMA.ADR

```

```

      CMP      R1,@#242
      BEQ      $50025
      JSR      PC.ERROR
$50025:
      CMP      R2,@#402
      BEQ      $50026
      JSR      PC.ERROR
$50026:
      CMP      R3,@#404
      BEQ      $50027
      JSR      PC.ERROR
$50027:
      TST      R4
      BEQ      $50030
      JSR      PC.ERROR
$50030:
      TST      R5
      BEQ      $50031
      JSR      PC.ERROR
$50031:
      JMP      XL3DR

```

L3DR

```

1498 .SBTTL L3DR
1499 ;*****
1500 ;
1501 ; L3DR TEST
1502 ; THIS TEST USES L3D3, L3D5, AND L3D7 TO TEST L3DR.
1503 ;
1504 ;*****
1505
1506 ROUTINE XL3DR
1507 002354 XL3DR:
1508 002354 000240 INLINE <NOP> NOP
1509 002356 000004 INLINE <SCOPE> SCOPE
1510 002360 012767 000020 176016 LET $TIMES := #20 ;ITERATION COUNT
1511 002366 012767 000002 176074 LET COUNT := #2 MOV #20,$TIMES
1512 002374 REPEAT MOV #2,COUNT
1513 ;
1514 ; SET UP THE GPR'S AND DO
1515 ; THE CIS INSTRUCTION
1516 ;
1517
1518 002374 CALL CLRREG
1519 002374 004767 054040 JSR PC,CLRREG
1520 002400 026727 176064 000002 IF COUNT EQ #2 THEN
1521 002406 001005 CMP COUNT,#2
1522 002410 LET R3 := #ADR.A BNE $50035
1523 002414 012703 002332 MOV #ADR.A,R3
1524 002414 000257 INLINE <CCC> CCC
1525 002416 076063 INLINE <L3D3> L3D3
1526 002420 ELSE BR $50036
1527 002422 000404 $50035:
1528 002422 LET R5 := #ADR.A MOV #ADR.A,R5
1529 002426 012705 002332 INLINE <SCC> SCC
1530 002430 076065 INLINE <L3D5> L3D5
1531 002432 ENDIF $50036:
1532 002432 LET CC := PSW MOV PSW,CC
1533 002432 016767 175340 176060 LET CC := CC CLR.BY #177760 BIC #177760,CC
1534 002440 042767 177760 176052 LET ERR := #1 ;CODE FOR INSTRUCTION FAILURE
1535 002446 012767 000001 176076 MOV #1,ERR
1536
1537 ;
1538 ; CHECK RESULTS AND REPORT ERRORS

```

L3DR

1534							
1535							
1536	002454				IF COUNT EQ #2 THEN		
	002454	026727	176010	000002		CMP	COUNT,#2
	002462	001006				BNE	\$50037
1537	002464				IF CC NE #0 THEN		
	002464	005767	176030			TST	CC
	002470	001402				BEQ	\$50040
1538	002472				CALL ERROR		
	002472	004767	055202			JSR	PC.ERROR
1539	002476				ENDIF		
	002476					\$50040:	
1540	002476				ELSE		
	002476	000406				BR	\$50041
	002500					\$50037:	
1541	002500				IF CC NE #17 THEN		
	002500	026727	176014	000017		CMP	CC,#17
	002506	001402				BEQ	\$50042
1542	002510				CALL ERROR		
	002510	004767	055164			JSR	PC.ERROR
1543	002514				ENDIF		
	002514					\$50042:	
1544	002514				ENDIF		
	002514					\$50041:	
1545	002514				IF R0 NE ALPHA THEN		
	002514	020067	177620			CMP	R0,ALPHA
	002520	001402				BEQ	\$50043
1546	002522				CALL ERROR		
	002522	004767	055152			JSR	PC.ERROR
1547	002526				ENDIF		
	002526					\$50043:	
1548	002526				IF R1 NE ALPHA+2 THEN		
	002526	020167	177610			CMP	R1,ALPHA+2
	002532	001402				BEQ	\$50044
1549	002534				CALL ERROR		
	002534	004767	055140			JSR	PC.ERROR
1550	002540				ENDIF		
	002540					\$50044:	
1551	002540				IF R2 NE BETA THEN		
	002540	020267	177600			CMP	R2,BETA
	002544	001402				BEQ	\$50045
1552	002546				CALL ERROR		
	002546	004767	055126			JSR	PC.ERROR
1553	002552				ENDIF		
	002552					\$50045:	
1554	002552				IF R3 NE BETA+2 THEN		
	002552	020367	177570			CMP	R3,BETA+2
	002556	001402				BEQ	\$50046
1555	002560				CALL ERROR		
	002560	004767	055114			JSR	PC.ERROR
1556	002564				ENDIF		
	002564					\$50046:	
1557	002564				IF R4 NE GAMMA THEN		
	002564	020467	177560			CMP	R4,GAMMA
	002570	001402				BEQ	\$50047
1558	002572				CALL ERROR		
	002572	004767	055102			JSR	PC.ERROR

L3DR

1559	002576		ENDIF				
	002576					\$50047:	
1560	002576		IF R5 NE GAMMA+2 THEN				
	002576	020567					CMP R5,GAMMA+2
	002602	001402					BEQ \$50050
1561	002604		CALL ERROR				
	002604	004767					JSR PC,ERROR
1562	002610		ENDIF				
	002610					\$50050:	
1563	002610		LET COUNT := COUNT - #1				
	002610	005367					DEC COUNT
1564	002614		UNTIL COUNT EQ #0				
	002614	005767					TST COUNT
	002620	001265					BNE \$50034
1565							
1566							
1567			;; TEST THE SPECIAL CASE OF USING R7				
1568			;-				
1569							
1570	002622		INLINE <L3D7>				
	002622	076067				L3D7	
1571	002624		INLINE <.WORD 240> ;NOP				.WORD 240
	002624	000240					
1572	002626		INLINE <.WORD 240> ;NOP				.WORD 240
	002626	000240					
1573	002630		INLINE <.WORD 402> ;BR TO ERROR CALL				.WORD 402
	002630	000402					
1574	002632		INLINE <.WORD 403> ;BR OVER ERROR CALL				.WORD 403
	002632	000403					
1575	002634		INLINE <.WORD 240> ;NOP				.WORD 240
	002634	000240					
1576	002636		CALL ERROR				
	002636	004767					JSR PC,ERROR
1577	002642		IF R0 NE @#240 THEN				
	002642	020037					CMP R0,@#240
	002646	001402					BEQ \$50051
1578	002650		CALL ERROR				
	002650	004767					JSR PC,ERROR
1579	002654		ENDIF				
	002654					\$50051:	
1580	002654		IF R1 NE @#242 THEN				
	002654	020137					CMP R1,@#242
	002660	001402					BEQ \$50052
1581	002662		CALL ERROR				
	002662	004767					JSR PC,ERROR
1582	002666		ENDIF				
	002666					\$50052:	
1583	002666		IF R2 NE @#240 THEN				
	002666	020237					CMP R2,@#240
	002672	001402					BEQ \$50053
1584							
1585	002674		CALL ERROR				
	002674	004767					JSR PC,ERROR
1586	002700		ENDIF				
	002700					\$50053:	
1587	002700		IF R3 NE @#242 THEN				
	002700	020337					CMP R3,@#242

L3DR

1588	002704	001402			BEQ	\$50054
	002706		CALL ERROR			
	002706	004767	054766		JSR	PC.ERROR
1589	002712		ENDIF			
	002712				\$50054:	
1590	002712		IF R3 NE @#242 THEN			
	002712	020337	000242		CMP	R3,@#242
	002716	001402			BEQ	\$50055
1591	002720		CALL ERROR			
	002720	004767	054754		JSR	PC.ERROR
1592	002724		ENDIF			
	002724				\$50055:	
1593	002724		IF R4 NE @#402 THEN			
	002724	020437	000402		CMP	R4,@#402
	002730	001402			BEQ	\$50056
1594	002732		CALL ERROR			
	002732	004767	054742		JSR	PC.ERROR
1595	002736		ENDIF			
	002736				\$50056:	
1596	002736		IF R5 NE @#404 THEN			
	002736	020537	000404		CMP	R5,@#404
	002742	001402			BEQ	\$50057
1597	002744		CALL ERROR			
	002744	004767	054730		JSR	PC.ERROR
1598	002750		ENDIF			
	002750				\$50057:	
1599	002750		INLINE <JMP XMOVC>			
	002750	000167	000000		JMP	XMOVC
1600						
1601						

MOVC

```

1603 .SBTTL MOVC
1604 ;*****
1605 ;
1606 ; MOVC TEST
1607 ;
1608 ;*****
1609 ;
1610 002754 ROUTINE XMOVC
1611 002754 XMOVC:
1612 002754 000240 INLINE <NOP> NOP
1613 002756 000004 INLINE <SCOPE> SCOPE
1614 002760 012767 000020 175416 LET $TIMES := #20 ;ITERATION COUNT MOV #20,$TIMES
1615 002766 016767 001306 175474 LET COUNT := TMOVC ;GET TEST COUNT MOV TMOVC,COUNT
1616 002774 012767 004302 175542 LET TSTCAS := #TMOVC*2 ;POINT TO FIRST TEST CASE MOV #TMOVC*2,TSTCAS
1617 003002 003002 INLINE <#5030:> #5030:
1618 ;*
1619 ; GET TEST DATA FROM TABLE
1620 ;-
1621 ;
1622 003002 LET R0 := TSTCAS
1623 003006 016700 175536 LET DATA00 := (R0) ;SRC.LEN MOV TSTCAS,R0
1624 003012 012067 175550 LET DATA01 := (R0) ;SRC.ADR MOV (R0),DATA00
1625 003016 012067 175546 LET DATA02 := (R0) ;DST.LEN MOV (R0),DATA01
1626 003022 012067 175544 LET DATA03 := (R0) ;DST.ADR MOV (R0),DATA02
1627 003026 012067 175540 LET DATA04 := (R0) ;FILL CHAR. MOV (R0),DATA03
1628 003032 012067 175536 LET DATA05 := (R0) ;SRC.BUFF.STATE MOV (R0),DATA04
1629 003036 011067 175534 LET DATA06 := (R0) ;DST.BUFF.STATE MOV (R0),DATA06
1630 ;
1631 ;*
1632 ; SET UP THE GPR'S
1633 ; EXPECTED GPR'S AND CONDITION CODES,
1634 ; AND BUFFERS.
1635 ;-
1636 ;
1637 003042 LET R0 := DATA00 MOV DATA00,R0
1638 003046 016700 175514 LET R1 := DATA01 MOV DATA01,R1
1639 003052 016701 175512 LET R2 := DATA02 MOV DATA02,R2
1640 003056 016702 175510 LET R3 := DATA03 MOV DATA03,R3
1641 003056 016703 175506

```



```

MOVC
1641 003062          LET R4 := DATA04
      003062 016704 175504
1642 003066          IF #1 SETIN COUNT THEN
      003066 032767 000001 175374
      003074 001403
1643 003076          LET R5 := #177777
      003076 012705 177777
1644 003102          ELSE
      003102 000401
      003104
1645 003104          LET R5 := #0
      003104 005005
1646 003106          ENDIF
      003106
1647 003106          INLINE <CMP R0,R2>
      003106 020002
1648 003110          LET CC.EXP := PSW
      003110 016767 174662 175404
1649 003116          LET CC.EXP := CC.EXP CLR.BY #177760
      003116 042767 177760 175376
1650 003124          IF #CBIT SETIN CC.EXP OR #ZBIT SETIN CC.EXP THEN
      003124 032767 000001 175370
      003132 001004
      003134 032767 000004 175360
      003142 001403
      003144
1651 003144          LET R0.EXP := #0
      003144 005067 175354
1652 003150          ELSE
      003150 000404
      003152
1653 003152          LET R0.EXP := R0 - R2
      003152 010067 175346
      003156 160267 175342
      ;SRC.LEN - DST.LEN
1654 003162          ENDIF
      003162
1655 003162          LET R1.EXP := #0
      003162 005067 175340
1656 003166          LET R2.EXP := #0
      003166 005067 175336
1657 003172          LET R3.EXP := #0
      003172 005067 175334
1658 003176          LET R4.EXP := R4
      003176 010467 175332
1659 003202          LET R5.EXP := R5
      003202 010567 175330
1660 003206          LET BFADRS := R1
      003206 010167 175330
      ;SRC.ADR
1661 003212          IF DATA05 EQ #0 THEN
      003212 005767 175356
      003216 001003
1662 003220          CALL CLRBUF
      003220 004767 053232
1663 003224          ELSE
      003224 000411
      003226
1664 003226          IF DATA05 EQ #177777 THEN

```

MOV DATA04,R4

BIT #1,COUNT  
BEQ \$50062

MOV #177777,R5

BR \$50063

\$50062:

CLR R5

\$50063:

CMP R0,R2

MOV PSW,CC.EXP

BIC #177760,CC.EXP

BIT #CBIT,CC.EXP  
BNE \$50064  
BIT #ZBIT,CC.EXP  
BEQ \$50065

\$50064:

CLR R0.EXP

BR \$50066

\$50065:

MOV R0,R0.EXP  
SUB R2,R0.EXP

\$50066:

CLR R1.EXP

CLR R2.EXP

CLR R3.EXP

MOV R4,R4.EXP

MOV R5,R5.EXP

MOV R1,BFADRS

TST DATA05  
BNE \$50067

JSR PC,CLRBUF

BR \$50070

\$50067:

MOVC

```

003226 026727 175342 177777
003234 001903
1665 003236          CALL SETBUF          CMP      DATA05,#177777
003236 004767 053256          ELSE          BNE      $50071
1666 003242          CALL SETBUF          JSR      PC,SETBUF
003242 000402          ELSE          BR       $50072
003244          CALL ALTBUF          $50071:
1667 003244          CALL ALTBUF          JSR      PC,ALTBUF
003244 004767 053314          ENDIF
1668 003250          ENDIF          $50072:
003250          ENDIF          $50070:
1670 003250          LET BFADRS := R3          ;DST.ADR
003250 010367 175266          MOV      R3,BFADRS
1671 003254          IF DATA06 EQ #0 THEN
003254 005767 175316          CALL CLRBUF
003260 001003          ELSE
1672 003262          CALL CLRBUF          JSR      PC,CLRBUF
003262 004767 053170          ELSE          BR       $50074
1673 003266          ELSE          $50073:
003266 000411          IF DATA06 EQ #177777 THEN
003270          IF DATA06 EQ #177777 THEN
1674 003270 026727 175302 177777          CALL SETBUF
003270 001003          CALL SETBUF          BNE      $50075
1675 003300          CALL SETBUF          JSR      PC,SETBUF
003300 004767 053214          ELSE          BR       $50076
1676 003304          ELSE          $50075:
003304 000402          CALL ALTBUF          JSR      PC,ALTBUF
003306          CALL ALTBUF          $50076:
1677 003306          CALL ALTBUF          $50074:
003306 004767 053252          ENDIF
1678 003312          ENDIF
003312          ENDIF
1679 003312          IF DATA02 EQ #0 THEN
003312          IF DATA02 EQ #0 THEN
1680 003312          IF DATA02 EQ #0 THEN
003312 005767 175250          LET BF.EXP := (R3)
003316 001003          ELSE
1681 003320          LET BF.EXP := (R3)          MOV      (R3),BF.EXP
003320 011367 175214          ELSE          BR       $50100
1682 003324          ELSE          $50077:
003324 000402          LET BF.EXP := (R1)
003326          LET BF.EXP := (R1)
1683 003326          LET BF.EXP := (R1)          MOV      (R1),BF.EXP
003326 011167 175206          ENDIF
1684 003332          ENDIF          $50100:
003332
1685
1686          ; DO THE CIS INSTRUCTION
1687          ;
1688          ;
1689          ;
1690 003332          INLINE <MOVC>
003332 076030          MOVC
1691

```

MOVC

```

1692
1693          ; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS
1694          ;
1695
1696 003334          LET CC := PSW
1697 003334 016767 174436 175156          MOV      PSW,CC
1698 003342          LET CC := CC CLR.BY #177760
1699 003342 042767 177760 175150          BIC      #177760,CC
1700 003350          IF CC NE CC.EXP THEN
1701 003350 026767 175144 175144          CMP      CC,CC.EXP
1702 003356 001402          BEQ      $50101
1703 003360          CALL ERROR
1704 003360 004767 054314          JSR      PC,ERROR
1705 003364          ENDIF
1706 003364          IF R0 NE R0.EXP THEN
1707 003364 020067 175134          CMP      R0,R0.EXP
1708 003370 001402          BEQ      $50102
1709 003372          CALL ERROR
1710 003372 004767 054302          JSR      PC,ERROR
1711 003376          ENDIF
1712 003376          IF R1 NE R1.EXP THEN
1713 003376 020167 175124          CMP      R1,R1.EXP
1714 003402 001402          BEQ      $50103
1715 003404          CALL ERROR
1716 003404 004767 054270          JSR      PC,ERROR
1717 003410          ENDIF
1718 003410          IF R2 NE R2.EXP THEN
1719 003410 020267 175114          CMP      R2,R2.EXP
1720 003414 001402          BEQ      $50104
1721 003416          CALL ERROR
1722 003416 004767 054256          JSR      PC,ERROR
1723 003422          ENDIF
1724 003422          IF R3 NE R3.EXP THEN
1725 003422 020367 175104          CMP      R3,R3.EXP
1726 003426 001402          BEQ      $50105
1727 003430          CALL ERROR
1728 003430 004767 054244          JSR      PC,ERROR
1729 003434          ENDIF
1730 003434          IF R4 NE R4.EXP THEN
1731 003434 020467 175074          CMP      R4,R4.EXP
1732 003440 001402          BEQ      $50106
1733 003442          CALL ERROR
1734 003442 004767 054232          JSR      PC,ERROR
1735 003446          ENDIF
1736 003446          IF R5 NE R5.EXP THEN
1737 003446 020567 175064          CMP      R5,R5.EXP
1738 003452 001402          BEQ      $50107
1739 003454          CALL ERROR
1740 003454 004767 054220          JSR      PC,ERROR
1741 003460          ENDIF
1742 003460

```

MOVC

```

1719
1720
1721      ;*
1722      ; CHECK ENTIRE DESTINATION BUFFER FOR CORRECT RESULTS
1723      ;
1724 003460      LET R0 := DATA01          ;SRC.ADR
1725 003460 016700 175100      MOV      DATA01,R0
1726 003464 016701 175100      LET R1 := DATA03          ;DST.ADR
1727 003470 005067 174762      MOV      DATA03,R1
1728 003474 026767 175062 175064      LET ERRFLG := #0
1729 003474 103417      CLR      ERRFLG
1730 003504      IF DATA00 HIS DATA02 THEN
1731 003504 016703 175056      CMP      DATA00,DATA02
1732 003510      LET R3 := DATA02          ;BLO      $50110
1733 003510      WHILE R3 NE #0 AND ERRFLG EQ #0 DO
1734 003510 005703      $50111:      TST      R3
1735 003512 001412      BEQ      $50112
1736 003514 005767 174736      TST      ERRFLG
1737 003520 001007      BNE      $50112
1738 003522      IFB BF.EXP NE (R1)+ THEN
1739 003522 126721 175012      CMPB     BF.EXP,(R1)+
1740 003526 001402      BEQ      $50113
1741 003530      CALL ERROR
1742 003530 004767 054144      JSR      PC.ERROR
1743 003534      ENDIF
1744 003534      LET R3 := R3 - #1
1745 003534 005303      $50113:      DEC      R3
1746 003536      ENDDO
1747 003536 000764      BR      $50111
1748 003540      ELSE
1749 003540 000441      $50112:      BR      $50114
1750 003542      LET R3 := DATA00          ;SRC.LEN
1751 003542 016703 175014      $50110:      MOV      DATA00,R3
1752 003546      WHILE R3 NE #0 AND ERRFLG EQ #0 DO
1753 003546 005703      $50115:      TST      R3
1754 003550 001412      BEQ      $50116
1755 003552 005767 174700      TST      ERRFLG
1756 003556 001007      BNE      $50116
1757 003560      IFB BF.EXP NE (R1)+ THEN
1758 003560 126721 174754      CMPB     BF.EXP,(R1)+
1759 003564 001402      BEQ      $50117
1760 003566      CALL ERROR
1761 003566 004767 054106      JSR      PC.ERROR
1762 003572      ENDIF
1763 003572      LET R3 := R3 - #1
1764 003572 005303      $50117:      DEC      R3
1765 003574      ENDDO
1766 003574 000764      BR      $50115
1767 003576      $50116:

```

MOVC						
1743	003576			LET R3 := DATA02	DATA00	
	003576	016703	174764			MOV DATA02,R3
	003602	166703	174754			SUB DATA00,R3
1744	003606			LET R1 := DATA03	DATA00	
	003606	016701	174756			MOV DATA03,R1
	003612	066701	174744			ADD DATA00,R1
1745	003616			REPEAT		
	003616					\$50120:
1746	003616			IFB (R1) NE DATA04 THEN		
	003616	122167	174750			CMPB (R1),DATA04
	003622	001402				BEQ \$50121
1747	003624			CALL ERROR		
	003624	004767	054050			JSR PC,ERROR
1748	003630			ENDIF		
	003630					\$50121:
1749	003630			LET R3 := R3 - #1		
	003630	005303				DEC R3
1750	003632			UNTIL R3 EQ #0 OR ERRFLG NE #0		
	003632	005703				TST R3
	003634	001403				BEQ \$50122
	003636	005767	174614			TST ERRFLG
	003642	001765				BEQ \$50120
	003644					\$50122:
1751	003644			ENDIF		
	003644					\$50114:
1752	003644			LET ERRFLG := #0		
	003644	005067	174606			CLR ERRFLG
1753	003650			IF DATA02 LO #BUFLEN THEN		
	003650	026727	174712		000100	CMP DATA02,#BUFLEN
	003656	103040				BHIS \$50123
1754	003660			IF DATA06 EQ #1 THEN		
	003660	026727	174712		000001	CMP DATA06,#1
	003666	001004				BNE \$50124
1755	003670			LET BF.EXP := #125252		
	003670	012767	125252		174642	MOV #125252,BF.EXP
1756	003676			ELSE		
	003676	000403				BR \$50125
	003700					\$50124:
1757	003700			LET BF.EXP := DATA06		
	003700	016767	174672		174632	MOV DATA06,BF.EXP
1758	003706			ENDIF		
	003706					\$50125:
1759	003706			LET R0 := DATA02	DATA03	
	003706	016700	174654			MOV DATA02,R0
	003712	066700	174652			ADD DATA03,R0
1760	003716			LET TEMPO := DATA03	#BUFLEN	
	003716	016767	174646		174552	MOV DATA03,TEMPO
	003724	062767	000100		174544	ADD #BUFLEN,TEMPO
1761	003732			REPEAT		
	003732					\$50126:
1762	003732			IFB (R0) NE BF.EXP THEN		
	003732	122067	174602			CMPB (R0),BF.EXP
	003736	001402				BEQ \$50127
1763	003740			CALL ERROR		
	003740	004767	053734			JSR PC,ERROR
1764	003744			ENDIF		
	003744					\$50127:

MOVC

1765	003744			UNTIL R0 EQ TEMPO OR ERRFLG NE #0		
	003744	020067	174526			
	003750	001403				
	003752	005767	174500			
	003756	001765				
	003760					
1766	003760			ENDIF		
	003760					
1767	003760			LET TSTCAS := TSTCAS + #16		
	003760	062767	000016	174556		
1768	003766			LET COUNT := COUNT #1		
	003766	005367	174476			
1769				:UNTIL COUNT EQ #0		
1770	003772			IF COUNT NE #0 THEN		
	003772	005767	174472			
	003776	001402				
1771	004000			INLINE <JMP \$5030>		
	004000	000167	176776			
1772	004004			ENDIF		
	004004					
1773						
1774						
1775						
1776				: DO ONE IN-LINE CASE - MOVCI		
1777				:		
1778				:		
1779	004004			LET BFADRS := #BUFF1		
	004004	012767	063236	174530		
1780	004012			CALL SETBUF		
	004012	004767	052502			
1781	004016			LET BFADRS := #BUFF2		
	004016	012767	063336	174516		
1782	004024			CALL CLRBUF		
	004024	004767	052426			
1783	004030			LET R0 := #125252		
	004030	012700	125252			
1784	004034			LET R1 := #125252		
	004034	012701	125252			
1785	004040			LET R2 := #125252		
	004040	012702	125252			
1786	004044			LET R3 := #125252		
	004044	012703	125252			
1787	004050			LET R4 := #125252		
	004050	012704	125252			
1788	004054			LET R5 := #125252		
	004054	012705	125252			
1789	004060			LET ERRFLG := #0		
	004060	005067	174372			
1790	004064			INLINE <MOVCI>		
	004064	076130				
1791	004066			INLINE <.WORD TMOVCI>		
	004066	004624				
1792	004070			INLINE <.WORD TMOVCI+4>		
	004070	004630				
1793	004072			INLINE <.WORD 40>		
	004072	000040				
1794	004074			IFCOND CS THEN		

MOVC

1795	004074	103003			BCC	\$50132
	004076			IFCOND EQ THEN		
1796	004076	001002			BNE	\$50133
	004100			CALL ERROR		
1797	004100	004767	053574		JSR	PC.ERROR
	004104			ENDIF		
	004104				\$50133:	
1798	004104			ENDIF		
	004104				\$50132:	
1799	004104			IF R0 NE #125252 THEN		
	004104	020027	125252		CMP	R0,#125252
	004110	001402			BEQ	\$50134
1800	004112			CALL ERROR		
	004112	004767	053562		JSR	PC.ERROR
1801	004116			ENDIF		
	004116				\$50134:	
1802	004116			IF R1 NE #125252 THEN		
	004116	020127	125252		CMP	R1,#125252
	004122	001402			BEQ	\$50135
1803	004124			CALL ERROR		
	004124	004767	053550		JSR	PC.ERROR
1804	004130			ENDIF		
	004130				\$50135:	
1805	004130			IF R2 NE #125252 THEN		
	004130	020227	125252		CMP	R2,#125252
	004134	001402			BEQ	\$50136
1806	004136			CALL ERROR		
	004136	004767	053536		JSR	PC.ERROR
1807	004142			ENDIF		
	004142				\$50136:	
1808	004142			IF R3 NE #125252 THEN		
	004142	020327	125252		CMP	R3,#125252
	004146	001402			BEQ	\$50137
1809	004150			CALL ERROR		
	004150	004767	053524		JSR	PC.ERROR
1810	004154			ENDIF		
	004154				\$50137:	
1811	004154			IF R4 NE #125252 THEN		
	004154	020427	125252		CMP	R4,#125252
	004160	001402			BEQ	\$50140
1812	004162			CALL ERROR		
	004162	004767	053512		JSR	PC.ERROR
1813	004166			ENDIF		
	004166				\$50140:	
1814	004166			IF R5 NE #125252 THEN		
	004166	020527	125252		CMP	R5,#125252
	004172	001402			BEQ	\$50141
1815	004174			CALL ERROR		
	004174	004767	053500		JSR	PC.ERROR
1816	004200			ENDIF		
	004200				\$50141:	
1817	004200			IF BUFF2 NE #177777 THEN		
	004200	026727	057132 177777		CMP	BUFF2,#177777
	004206	001402			BEQ	\$50142
1818	004210			CALL ERROR		
	004210	004767	053464		JSR	PC.ERROR
1819	004214			ENDIF		

MOVC

```

1820 004214 026727 057120 020040 IF BUFF2+2 NE #20040 THEN $50142:
004214 001402                                CMP     BUFF2+2,#20040
1821 004224 004767 053450          CALL ERROR                                BEQ     $50143
004224 004767 053450          JSR     PC,ERROR
1822 004230 004230 004230          ENDIF
004230 004230 004230          $50143:
1823 004230 012700 063342          LET RO := #BUFF2+4                        MOV     #BUFF2+4,RO
1824 004234 012767 063336 174234    LET TEMPO := #BUFF2 + #BUFLN-2           MOV     #BUFF2,TEMPO
004234 062767 000076 174226          ADD     #BUFLN 2,TEMPO
1825 004250 004250 004250          REPEAT
004250 004250 005720 001402          IF (RO)+ NE #0 THEN $50144:
1826 004252 004254 004767 053420    CALL ERROR                                TST     (RO)+
004254 004767 053420          JSR     PC,ERROR                          BEQ     $50145
1827 004260 004260 004260          ENDIF
004260 004260 004260          $50145:
1828 004260 020067 174212          UNTIL RO EQ TEMPO OR ERRFLG NE #0       CMP     RO,TEMPO
004260 001403 004266 005767 174164    BEQ     $50146
004266 001766 004272 001766          TST     ERRFLG
004272 001766 004274 001766          BEQ     $50144
1829 004274 000167 000334          $50146:
004274 000167 000334          JMP     XMOVRC
1830 004274 000167 000334          INLINE <JMP XMOVRC>
1831 004274 000167 000334

```

```

1831
1832 ;+
1833 ; TABLE OF TEST CASES FOR MOVC
1834 ;-
1835

```

1836	004300	000017	TMOV: .WORD 15.	;	# OF TEST CASES FOR MOVC
1837					
1838	004302	000040	.WORD 40	;	SRC.LEN
1839	004304	063336	.WORD BUFF2	;	SRC.ADR
1840	004306	000000	.WORD 0	;	DST.LEN
1841	004310	063236	.WORD BUFF1	;	DST.ADR
1842	004312	000040	.WORD 40	;	FILL.CHAR.
1843	004314	000000	.WORD 0	;	SRC.BUFF.STATE
1844	004316	177777	.WORD -1	;	DST.BUFF.STATE
1845					
1846	004320	000050	.WORD 50	;	TEST CASE NUMBER 2
1847	004322	063336	.WORD BUFF2		
1848	004324	000020	.WORD 20		
1849	004326	063236	.WORD BUFF1		
1850	004330	000100	.WORD 100		
1851	004332	177777	.WORD -1		
1852	004334	000000	.WORD 0		
1853					
1854	004336	000045	.WORD 45	;	TEST CASE NUMBER 3
1855	004340	063336	.WORD BUFF2		
1856	004342	000030	.WORD 30		
1857	004344	063236	.WORD BUFF1		



MOVC

1858	004346	000007	.WORD	7	
1859	004350	000000	.WORD	0	
1860	004352	000001	.WORD	1	
1861					
1862	004354	000000	.WORD	0	;TEST CASE NUMBER 4
1863	004356	063336	.WORD	BUFF2	
1864	004360	000060	.WORD	60	
1865	004362	063236	.WORD	BUFF1	
1866	004364	000010	.WORD	10	
1867	004366	000001	.WORD	1	
1868	004370	000000	.WORD	0	
1869					
1870	004372	000046	.WORD	46	;TEST CASE NUMBER 5
1871	004374	063336	.WORD	BUFF2	
1872	004376	000132	.WORD	BUFLN+32	
1873	004400	063236	.WORD	BUFF1	
1874	004402	000077	.WORD	77	
1875	004404	177777	.WORD	-1	
1876	004406	000001	.WORD	1	
1877					
1878	004410	000000	.WORD	0	;TEST CASE NUMBER 6
1879	004412	063336	.WORD	BUFF2	
1880	004414	000015	.WORD	15	
1881	004416	063236	.WORD	BUFF1	
1882	004420	000060	.WORD	60	
1883	004422	000001	.WORD	1	
1884	004424	177777	.WORD	-1	
1885					
1886	004426	000043	.WORD	43	;TEST CASE NUMBER 7
1887	004430	063336	.WORD	BUFF2	
1888	004432	000070	.WORD	70	
1889	004434	063236	.WORD	BUFF1	
1890	004436	000370	.WORD	370	
1891	004440	000000	.WORD	0	
1892	004442	177777	.WORD	-1	
1893					
1894	004444	000060	.WORD	60	;TEST CASE NUMBER 8
1895	004446	063236	.WORD	BUFF1	
1896	004450	000000	.WORD	0	
1897	004452	063336	.WORD	BUFF2	
1898	004454	000200	.WORD	200	
1899	004456	177777	.WORD	-1	
1900	004460	000000	.WORD	0	
1901					
1902	004462	000120	.WORD	BUFLN+20	;TEST CASE NUMBER 9
1903	004464	063236	.WORD	BUFF1	
1904	004466	000100	.WORD	100	
1905	004470	063336	.WORD	BUFF2	
1906	004472	000002	.WORD	2	
1907	004474	000000	.WORD	0	
1908	004476	000001	.WORD	1	
1909					
1910	004500	000015	.WORD	15	;TEST CASE NUMBER 10
1911	004502	063236	.WORD	BUFF1	
1912	004504	000000	.WORD	0	
1913	004506	063336	.WORD	BUFF2	
1914	004510	000016	.WORD	16	

MOVC

```

1915 004512 000001 .WORD 1
1916 004514 000000 .WORD 0
1917
1918 004516 000077 .WORD 77 ;TEST CASE NUMBER 11
1919 004520 063236 .WORD BUFF1
1920 004522 000021 .WORD 21
1921 004524 063336 .WORD BUFF2
1922 004526 000300 .WORD 300
1923 004530 177777 .WORD -1
1924 004532 000001 .WORD 1
1925
1926 004534 000000 .WORD 0 ;TEST CASE NUMBER 12
1927 004536 063236 .WORD BUFF1
1928 004540 000050 .WORD 50
1929 004542 063336 .WORD BUFF2
1930 004544 000003 .WORD 3
1931 004546 000001 .WORD 1
1932 004550 177777 .WORD -1
1933
1934 004552 000014 .WORD 14 ;TEST CASE NUMBER 13
1935 004554 063236 .WORD BUFF1
1936 004556 000042 .WORD 42
1937 004560 063336 .WORD BUFF2
1938 004562 000030 .WORD 30
1939 004564 000000 .WORD 0
1940 004566 177777 .WORD -1
1941
1942 004570 000000 .WORD 0 ;TEST CASE NUMBER 14
1943 004572 063236 .WORD BUFF1
1944 004574 000053 .WORD 53
1945 004576 063336 .WORD BUFF2
1946 004600 000012 .WORD 12
1947 004602 000001 .WORD 1
1948 004604 000000 .WORD 0
1949
1950 004606 000027 .WORD 27 ;TEST CASE NUMBER 15
1951 004610 063236 .WORD BUFF1
1952 004612 000060 .WORD 60
1953 004614 063336 .WORD BUFF2
1954 004616 000014 .WORD 14
1955 004620 177777 .WORD -1
1956 004622 000001 .WORD 1
1957
1958
1959 004624 000002 TMOVCI: .WORD 2 ;TEST CASE FOR IN LINE
1960 004626 063236 .WORD BUFF1
1961 004630 000004 .WORD 4
1962 004632 063336 .WORD BUFF2
1963

```

MOVRC

```

1965 .SBTTL MOVRC
1966 ;*****
1967 ;
1968 ; MOVRC TEST
1969 ;
1970 ;*****
1971 ;
1972 ROUTINE XMOVRC
1973 004634 XMOVRC:
1974 004634 000240 INLINE <NOP> NOP
1975 004636 000004 INLINE <SCOPE> SCOPE
1976 004640 012767 000020 173536 LET $TIMES := #20 ;ITERATION COUNT MOV #20,$TIMES
1977 004646 016767 001274 173614 LET COUNT := TMOVRC ;GET TEST COUNT MOV TMOVRC,COUNT
1978 004654 012767 006150 173662 LET TSTCAS := #TMOVRC+2 ;POINT TO FIRST TEST CASE MOV #TMOVRC+2,TSTCAS
1979 004662 004662 INLINE <#5031:> $5031:
1980 ;*
1981 ; GET TEST DATA FROM TABLE
1982 ;-
1983 ;
1984 004662 LET R0 := TSTCAS
1985 004666 016700 173656 MOV TSTCAS,R0
1986 004666 012067 173670 LET DATA00 := (R0) ;SRC.LEN MOV (R0),DATA00
1987 004672 012067 173666 LET DATA01 := (R0) ;SRC.ADR MOV (R0),DATA01
1988 004676 012067 173664 LET DATA02 := (R0) ;DST.LEN MOV (R0),DATA02
1989 004702 012067 173662 LET DATA03 := (R0) ;DST.ADR MOV (R0),DATA03
1990 004706 012067 173660 LET DATA04 := (R0) ;FILL CHAR. MOV (R0),DATA04
1991 004712 012067 173656 LET DATA05 := (R0) ;SRC.BUFF.STATE MOV (R0),DATA05
1992 004716 011067 173654 LET DATA06 := (R0) ;DST.BUFF.STATE MOV (R0),DATA06
1993 ;*
1994 ; SET UP GPR'S, EXPECTED GPR'S AND
1995 ; CONDITION CODES, AND BUFFERS.
1996 ;
1997 ;
1998 004722 LET R0 := DATA00 MOV DATA00,R0
1999 004726 016700 173634 LET R1 := DATA01 MOV DATA01,R1
2000 004732 016701 173632 LET R2 := DATA02 MOV DATA02,R2
2001 004736 016702 173630 LET R3 := DATA03 MOV DATA03,R3
2002 004742 016703 173626 LET R4 := DATA04

```

MOVRC						
2003	004742	016704	173624			MOV DATA04,R4
	004746			IF #1 SETIN COUNT THEN		BIT #1,COUNT
	004746	032767	000001	173514		BEG \$50151
	004754	001403				
2004	004756			LET R5 := #177777		MOV #177777,R5
	004756	012705	177777			
2005	004762			ELSE		BR \$50152
	004762	000401				\$50151:
	004764			LET R5 := #0		CLR R5
2006	004764	005005				
	004764			ENDIF		\$50152:
2007	004766			INLINE <CMP R0,R2>		
	004766	020002				CMP R0,R2
2008	004766			LET CC.EXP := PSW		MOV PSW,CC.EXP
2009	004770			LET CC.EXP := CC.EXP CLR.BY #177760		BIC #177760,CC.EXP
	004770	016767	173002	173524		
2010	004776	042767	177760	173516		BIT #CBIT,CC.EXP
	004776			IF #CBIT SETIN CC.EXP OR #ZBIT SETIN CC.EXP THEN		BNE \$50153
2011	005004					BIT #ZBIT,CC.EXP
	005004	032767	000001	173510		BEG \$50154
	005004	001004				
	005014	032767	000004	173500		
	005022	001403				\$50153:
	005024			LET R0.EXP := #0		CLR R0.EXP
2012	005024	005067	173474			
	005024			ELSE		BR \$50155
2013	005030					\$50154:
	005030	000404		LET R0.EXP := R0 - R2		MOV R0,R0.EXP
	005032					SUB R2,R0.EXP
2014	005032	010067	173466			
	005032			ENDIF		\$50155:
	005036	160267	173462			CLR R1.EXP
2015	005042			LET R1.EXP := #0		CLR R2.EXP
	005042	005067	173460			CLR R3.EXP
2016	005042			LET R2.EXP := #0		MOV R4,R4.EXP
	005042			LET R3.EXP := #0		MOV R5,R5.EXP
2017	005046			LET R4.EXP := R4		MOV DATA01,BFADRS
	005046	005067	173456			
2018	005052			LET R5.EXP := R5		TST DATA05
	005052	005067	173454	LET BFADRS := DATA01		BNE \$50156
2019	005056			IF DATA05 EQ #0 THEN		
	005056	010467	173452			JSR PC,CLRBUF
2020	005062			CALL CLRBUF		
	005062	010567	173450			BR \$50157
2021	005066			ELSE		\$50156:
	005066	016767	173472	173446		CMP DATA05,#177777
2022	005074			IF DATA05 EQ #0 THEN		
	005074	005767	173474			
	005100	001003				
2023	005102			CALL CLRBUF		
	005102	004767	051350			
2024	005106			ELSE		
	005106	000411				
	005110			IF DATA05 EQ #177777 THEN		
2025	005110	026727	173460	177777		

```

MOVRC
2026 005116 001003                                CALL SETBUF                                BNE    $50160
      005120                                CALL SETBUF                                JSR    PC,SETBUF
2027 005120 004767 051374                          ELSE
      005124 000402                                BR     $50161
      005126                                $50160:
2028 005126                                CALL ALTBUF                                JSR    PC,ALTBUF
      005126 004767 051432                          ENDIF
2029 005132                                ENDIF                                $50161:
2030 005132                                ENDIF                                $50157:
2031 005132                                LET BFADRS := DATA03
      005132 016767 173432 173402                      MOV    DATA03,BFADRS
2032 005140                                IF DATA06 EQ #0 THEN
      005140 005767 173432                                TST   DATA06
      005144 001003                                BNE   $50162
2033 005146                                CALL CLRBUF                                JSR    PC,CLRBUF
      005146 004767 051304                          ELSE
2034 005152                                BR     $50163
      005152 000411                                $50162:
      005154                                IF DATA06 EQ #177777 THEN
2035 005154 026727 173416 177777                      CMP   DATA06,#177777
      005162 001003                                BNE   $50164
2036 005164                                CALL SETBUF                                JSR    PC,SETBUF
      005164 004767 051330                          ELSE
2037 005170                                BR     $50165
      005170 000402                                $50164:
      005172                                CALL ALTBUF                                JSR    PC,ALTBUF
2038 005172 004767 051366                          ENDIF
2039 005176                                ENDIF                                $50165:
2040 005176                                ENDIF                                $50163:
2041 005176                                LET BF.EXP := @DATA03
      005176 017767 173366 173334                      MOV    @DATA03,BF.EXP
2042
2043 ;*
2044 ; DO THE LIS INSTRUCTION
2045 ; -
2046
2047 005204                                INLINE <MOVRC>
      005204 076031                                MOVRC
2048
2049 ;*
2050 ; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS
2051 ; -
2052
2053 005206                                LET CC := PSW
      005206 016767 172564 173304                      MOV    PSW,CC
2054 005214                                LET CC := CC CLR.BY #177760
      005214 042767 177760 173276                      BIC   #177760,CC
2055 005222                                IF CC NE CC.EXP THEN
      005222 026767 173272 173272                      CMP   CC,CC.EXP
      005230 001402                                BEQ   $50166

```

MOVRC

2056	005232			CALL ERROR			
	005232	004767	052442			JSR	PC,ERROR
2057	005236			ENDIF			
	005236					\$50166:	
2058	005236			IF R0 NE R0.EXP THEN			
	005236	020067	173262			CMP	R0,R0.EXP
	005242	001402				BEQ	\$50167
2059	005244			CALL ERROR			
	005244	004767	052430			JSR	PC,ERROR
2060	005250			ENDIF			
	005250					\$50167:	
2061	005250			IF R1 NE R1.EXP THEN			
	005250	020167	173252			CMP	R1,R1.EXP
	005254	001402				BEQ	\$50170
2062	005256			CALL ERROR			
	005256	004767	052416			JSR	PC,ERROR
2063	005262			ENDIF			
	005262					\$50170:	
2064	005262			IF R2 NE R2.EXP THEN			
	005262	020267	173242			CMP	R2,R2.EXP
	005266	001402				BEQ	\$50171
2065	005270			CALL ERROR			
	005270	004767	052404			JSR	PC,ERROR
2066	005274			ENDIF			
	005274					\$50171:	
2067	005274			IF R3 NE R3.EXP THEN			
	005274	020367	173232			CMP	R3,R3.EXP
	005300	001402				BEQ	\$50172
2068	005302			CALL ERROR			
	005302	004767	052372			JSR	PC,ERROR
2069	005306			ENDIF			
	005306					\$50172:	
2070	005306			IF R4 NE R4.EXP THEN			
	005306	020467	173222			CMP	R4,R4.EXP
	005312	001402				BEQ	\$50173
2071	005314			CALL ERROR			
	005314	004767	052360			JSR	PC,ERROR
2072	005320			ENDIF			
	005320					\$50173:	
2073	005320			IF R5 NE R5.EXP THEN			
	005320	020567	173212			CMP	R5,R5.EXP
	005324	001402				BEQ	\$50174
2074	005326			CALL ERROR			
	005326	004767	052346			JSR	PC,ERROR
2075	005332			ENDIF			
	005332					\$50174:	
2076							
2077							
2078				;; CHECK ENTIRE DESTINATION BUFFER FOR CORRECT RESULTS			
2079				;-			
2080							
2081	005332			LET ERRFLG := #0			
	005332	005067	173120			CLR	ERRFLG
2082	005336			LET R1 := DATA03			
	005336	016701	173226			MOV	DATA03,R1
2083	005342			IF DATA00 HIS DATA02 THEN			
	005342	026767	173214 173216			CMP	DATA00,DATA02

MOVRC

2084	005350	103424					BLO	\$50175
	005352			LET R3 := DATA02				
2085	005352	016703	173210				MOV	DATA02,R3
	005356			LET R0 := DATA01 + DATA00 DATA02				
	005356	016700	173202				MOV	DATA01,R0
	005362	066700	173174				ADD	DATA00,R0
	005366	166700	173174				SUB	DATA02,R0
2086	005372			WHILE R3 NE #0 AND ERRFLG EQ #0 DO				
	005372					\$50176:		
	005372	005703					TST	R3
	005374	001411					BEQ	\$50177
	005376	005767	173054				TST	ERRFLG
	005402	001006					BNE	\$50177
2087	005404			IFB (R0) + NE (R1) + THEN				
	005404	122021					CMPB	(R0) + ,(R1) +
	005406	001402					BEQ	\$50200
2088	005410			CALL ERROR				
	005410	004767	052264				JSR	PC,ERROR
2089	005414			ENDIF				
	005414					\$50200:		
2090	005414			LET R3 := R3 - #1				
	005414	005303					DEC	R3
2091	005416			ENDDO				
	005416	000765					BR	\$50176
	005420					\$50177:		
2092	005420			ELSE				
	005420	000455					BR	\$50201
	005422					\$50175:		
2093	005422			REPEAT				
	005422					\$50202:		
2094	005422			IFB (R1) + NE DATA04 THEN				
	005422	122167	173144				CMPB	(R1) + ,DATA04
	005426	001402					BEQ	\$50203
2095	005430			CALL ERROR				
	005430	004767	052244				JSR	PC,ERROR
2096	005434			ENDIF				
	005434					\$50203:		
2097	005434			LET TEMPO := DATA03 + DATA02 - DATA00				
	005434	016767	173130				MOV	DATA03,TEMPO
	005442	066767	173120				ADD	DATA02,TEMPO
	005450	166767	173106				SUB	DATA00,TEMPO
2098	005456			UNTIL R1 EQ TEMPO OR ERRFLG NE #0				
	005456	020167	173014				CMP	R1,TEMPO
	005462	001403					BEQ	\$50204
	005464	005767	172766				TST	ERRFLG
	005470	001754					BEQ	\$50202
	005472					\$50204:		
2099	005472			LET R0 := DATA01				
	005472	016700	173066				MOV	DATA01,R0
2100	005476			LET R1 := DATA03 + DATA02 - DATA00				
	005476	016701	173066				MOV	DATA03,R1
	005502	066701	173060				ADD	DATA02,R1
	005506	166701	173050				SUB	DATA00,R1
2101	005512			LET TEMPO := DATA03 + DATA02				
	005512	016767	173052				MOV	DATA03,TEMPO
	005520	066767	173042				ADD	DATA02,TEMPO
2102	005526			WHILE R1 NE TEMPO AND ERRFLG NE #0 DO				

MOVRC

005526						\$50205:	
005526	020167	172744				CMP	R1,TEMPO
005532	001410					BEQ	\$50206
005534	005767	172716				TST	ERRFLG
005540	001405					BEQ	\$50206
2103	005542				IFB (R0) NE (R1) THEN		
	005542	122021					
	005544	001402				CMPB	(R0), (R1)
2104	005546				CALL ERROR	BEQ	\$50207
	005546	004767	052126				
2105	005552				ENDIF	JSR	PC,ERROR
	005552						
2106	005552				ENDDO	\$50207:	
	005552	000765				BR	\$50205
	005554					\$50206:	
2107	005554				ENDIF	\$50201:	
	005554						
2108	005554				IF DATA02 LO #BUFLEN THEN		
	005554	026727	173006	000100		CMP	DATA02, #BUFLEN
	005562	103025				BHIS	\$50210
2109	005564				LET R0 := DATA03 + DATA02		
	005564	016700	173000			MOV	DATA03, R0
	005570	066700	172772			ADD	DATA02, R0
2110	005574				REPEAT		
	005574					\$50211:	
2111	005574				IFB (R0) NE BF.EXP THEN		
	005574	122067	172740				
	005600	001402				CMPB	(R0), BF.EXP
2112	005602				CALL ERROR	BEQ	\$50212
	005602	004767	052072				
2113	005606				ENDIF	JSR	PC,ERROR
	005606						
2114	005606				LET TEMPO := DATA03 + #BUFLEN	\$50212:	
	005606	016767	172756	172662			
	005614	062767	000100	172654		MOV	DATA03, TEMPO
2115	005622				UNTIL R0 EQ TEMPO OR ERRFLG NE #0	ADD	#BUFLEN, TEMPO
	005622	020067	172650				
	005626	001403				CMP	R0, TEMPO
	005630	005767	172622			BEQ	\$50213
	005634	001757				TST	ERRFLG
	005636					BEQ	\$50211
2116	005636				ENDIF	\$50213:	
	005636					\$50210:	
2117	005636				LET TSTCAS := TSTCAS + #16		
	005636	062767	000016	172700		ADD	#16, TSTCAS
2118	005644				LET COUNT := COUNT + #1		
	005644	005367	172620			DEC	COUNT
2119					:UNTIL COUNT EQ #0		
2120	005650				IF COUNT NE #0 THEN		
	005650	005767	172614				
	005654	001402				TST	COUNT
2121	005656				INLINE <JMP \$5031>	BEQ	\$50214
	005656	000167	177000				
2122	005662				ENDIF	JMP	\$5031
	005662						
2123						\$50214:	
2124							



MOVRC

```

2125
2126          ; DO ONE INLINE CASE - MOVRCI
2127          ;
2128
2129 005662          LET BFADRS := #BUFF1
2130 005662 012767 063236 172652          MOV      #BUFF1,BFADRS
2131 005670          CALL SETBUF
2132 005670 004767 050624          JSR      PC,SETBUF
2133 005674          LET BFADRS := #BUFF2
2134 005674 012767 063336 172640          MOV      #BUFF2,BFADRS
2135 005702          CALL CLRBUF
2136 005702 004767 050550          JSR      PC,CLRBUF
2137 005706          LET R0 := #125252
2138 005706 012700 125252          MOV      #125252,R0
2139 005712          LET R1 := #125252
2140 005712 012701 125252          MOV      #125252,R1
2141 005716          LET R2 := #125252
2142 005716 012702 125252          MOV      #125252,R2
2143 005722          LET R3 := #125252
2144 005722 012703 125252          MOV      #125252,R3
2145 005726          LET R4 := #125252
2146 005726 012704 125252          MOV      #125252,R4
2147 005732          LET R5 := #125252
2148 005732 012705 125252          MOV      #125252,R5
2149 005736          LET ERRFLG := #0
2150 005736 005067 172514          CLR      ERRFLG
2151 005742          INLINE <MOVRCI>
2152 005742 076131          MOVRCI
2153 005744          INLINE <.WORD      TMVRCI>          ;SRC.DSCR.PTR
2154 005744 006240          .WORD      TMVRCI
2155 005746          INLINE <.WORD      TMVRCI+4>          ;DST.DSCR.PTR
2156 005746 006244          .WORD      TMVRCI+4
2157 005750          INLINE <.WORD      40>          ;FILL CHARACTER
2158 005750 000040          .WORD      40
2159 005752          IFCOND CS THEN
2160 005752 103003          BCC      $50215
2161 005754          IFCOND EQ THEN
2162 005754 001002          BNE      $50216
2163 005756          CALL ERROR
2164 005756 004767 051716          JSR      PC,ERROR
2165 005762          ENDIF
2166 005762          ENDIF
2167 005762          IF R0 NE #125252 THEN
2168 005762 020027 125252          CMP      R0,#125252
2169 005766 001402          BEQ      $50217
2170 005770          CALL ERROR
2171 005770 004767 051704          JSR      PC,ERROR
2172 005774          ENDIF
2173 005774          IF R1 NE #125252 THEN
2174 005774 020127 125252          CMP      R1,#125252
2175 006000 001402          BEQ      $50220
2176 006002          CALL ERROR
2177 006002 004767 051672          JSR      PC,ERROR
2178 006006          ENDIF

```

MOVRC

2155	006006			IF R2 NE #125252 THEN	\$50220:		
	006006	020227	125252			CMP	R2,#125252
	006012	001402				BEQ	\$50221
2156	006014			CALL ERROR			
	006014	004767	051660			JSR	PC,ERROR
2157	006020			ENDIF	\$50221:		
	006020						
2158	006020			IF R3 NE #125252 THEN			
	006020	020327	125252			CMP	R3,#125252
	006024	001402				BEQ	\$50222
2159	006026			CALL ERROR			
	006026	004767	051646			JSR	PC,ERROR
2160	006032			ENDIF	\$50222:		
	006032						
2161	006032			IF R4 NE #125252 THEN			
	006032	020427	125252			CMP	R4,#125252
	006036	001402				BEQ	\$50223
2162	006040			CALL ERROR			
	006040	004767	051634			JSR	PC,ERROR
2163	006044			ENDIF	\$50223:		
	006044						
2164	006044			IF R5 NE #125252 THEN			
	006044	020527	125252			CMP	R5,#125252
	006050	001402				BEQ	\$50224
2165	006052			CALL ERROR			
	006052	004767	051622			JSR	PC,ERROR
2166	006056			ENDIF	\$50224:		
	006056						
2167	006056			IF BUFF2 NE #20040 THEN			
	006056	026727	055254 020040			CMP	BUFF2,#20040
	006064	001402				BEQ	\$50225
2168	006066			CALL ERROR			
	006066	004767	051606			JSR	PC,ERROR
2169	006072			ENDIF	\$50225:		
	006072						
2170	006072			IF BUFF2+2 NE #177777 THEN			
	006072	026727	055242 177777			CMP	BUFF2+2,#177777
	006100	001402				BEQ	\$50226
2171	006102			CALL ERROR			
	006102	004767	051572			JSR	PC,ERROR
2172	006106			ENDIF	\$50226:		
	006106						
2173	006106			LET ERRFLG := #0			
	006106	005067	172344			CLR	ERRFLG
2174	006112			LET R0 := #BUFF2+4			
	006112	012700	063342			MOV	#BUFF2+4,R0
2175	006116			REPEAT	\$50227:		
	006116						
2176	006116			IF (R0) NE #0 THEN			
	006116	005720				TST	(R0)
	006120	001402				BEQ	\$50230
2177	006122			CALL ERROR			
	006122	004767	051552			JSR	PC,ERROR
2178	006126			ENDIF	\$50230:		
	006126						
2179	006126			UNTIL R0 EQ #BUFF2*BUFLN-2 OR ERRFLG NE #0			

MOVRC

```

006126 020027 063434
006132 001403
006134 005767 172316
006140 001766
006142
2180 006142 000167 000102          INLINE <JMP XMOVTC>
006142
2181
2182
2183
2184
2185
2186 006146 000004          TMOVRC: .WORD 4          ;#OF TEST CASES FOR MOVRC
2187
2188 006150 000060          .WORD 60          ;SRC.LEN
2189 006152 063336          .WORD BUFF2      ;SRC.ADR
2190 006154 000025          .WORD 25         ;DST.LEN
2191 006156 063236          .WORD BUFF1      ;DST.ADR
2192 006160 000040          .WORD 40         ;FILL.CHAR.
2193 006162 000000          .WORD 0          ;SRC.BUFF.STATE
2194 006164 000001          .WORD 1          ;DST.BUFF.STATE
2195
2196 006166 000017          .WORD 17         ;TEST CASE NUMBER 2
2197 006170 063336          .WORD BUFF2
2198 006172 000050          .WORD 50
2199 006174 063236          .WORD BUFF1
2200 006176 000100          .WORD 100
2201 006200 000001          .WORD 1
2202 006202 000000          .WORD 0
2203
2204 006204 000030          .WORD 30         ;TEST CASE NUMBER 3
2205 006206 063236          .WORD BUFF1
2206 006210 000012          .WORD 12
2207 006212 063336          .WORD BUFF2
2208 006214 000077          .WORD 77
2209 006216 000000          .WORD 0
2210 006220 177777          .WORD -1
2211
2212 006222 000022          .WORD 22         ;TEST CASE NUMBER 4
2213 006224 063236          .WORD BUFF1
2214 006226 000037          .WORD 37
2215 006230 063236          .WORD BUFF1
2216 006232 000020          .WORD 20
2217 006234 177777          .WORD 1
2218 006236 000001          .WORD 1
2219
2220 006240 000002          TMVRCI: .WORD 2          ;TEST CASE FOR IN LINE
2221 006242 063236          .WORD BUFF1
2222 006244 000004          .WORD 4
2223 006246 063336          .WORD BUFF2
2224

```

```

CMP      R0,#BUFF2*BUFLN-2
BEQ      $50231
TST      ERRFLG
BEQ      $50227
$50231:
JMP      XMOVTC

```

MOVTC

```

2226 .SBTTL MOVTC
2227 ;*****
2228 ;
2229 ; MOVTC TEST
2230 ;
2231 ;*****
2232 ;
2233 006250 ROUTINE XMOVTC
2234 006250 XMOVTC:
2235 006250 000240 INLINE <NOP> NOP
2236 006252 000004 INLINE <SCOPE> SCOPE
2237 006254 012767 000020 172122 LET $TIMES := #20 ;ITERATION COUNT MOV #20,$TIMES
2238 006262 016767 001320 172200 LET COUNT := TMOVTC ;GET TEST COUNT MOV TMOVTC,COUNT
2239 006270 012767 007610 172246 LET TSTCAS := #TMOVTC+2 ;POINT TO FIRST TEST CASE MOV #TMOVTC+2,TSTCAS
2240 006276 $5032:
2241 ;*
2242 ; GET TEST DATA FROM TABLE
2243 ;-
2244 ;
2245 006276 LET RO := TSTCAS
2246 006302 016700 172242 MOV TSTCAS,RO
2247 006306 012067 172254 LET DATA00 := (RO)+ ;SRC.LEN MOV (RO)+,DATA00
2248 006306 012067 172252 LET DATA01 := (RO)+ ;SRC.ADR MOV (RO)+,DATA01
2249 006312 012067 172250 LET DATA02 := (RO)+ ;DST.LEN MOV (RO)+,DATA02
2250 006316 012067 172246 LET DATA03 := (RO)+ ;DST.ADR MOV (RO)+,DATA03
2251 006322 012067 172244 LET DATA04 := (RO)+ ;FILL CHAR. MOV (RO)+,DATA04
2252 006326 012067 172242 LET DATA05 := (RO)+ ;SRC.BUFF.STATE MOV (RO)+,DATA05
2253 006332 011067 172240 LET DATA06 := (RO) ;DST.BUFF.STATE MOV (RO),DATA06
2254 ;*
2255 ; SET UP GPR'S,
2256 ; EXPECTED GPR'S AND CONDITION CODES,
2257 ; AND BUFFERS.
2258 ;-
2259 ;
2260 006336 LET RO := DATA00 MOV DATA00,RO
2261 006342 016700 172220 LET R1 := DATA01 MOV DATA01,R1
2262 006346 016701 172216 LET R2 := DATA02 MOV DATA02,R2
2263 006352 016702 172214 LET R3 := DATA03 MOV DATA03,R3
2264 006352 016703 172212

```

MOVTC

2264	006356			LET R4 := DATA04		
	006356	016704	172210		MOV	DATA04,R4
2265	006362			LET R5 := #TRANS		
	006362	012705	063436		MOV	#TRANS,R5
2266	006366			INLINE <CMP R0,R2>		
	006366	020002			CMP	R0,R2
2267	006370			LET CC.EXP := PSW		
	006370	016767	171402 172124		MOV	PSW,CC.EXP
2268	006376			LET CC.EXP := CC.EXP CLR.BY #177760		
	006376	042767	177760 172116		BIC	#177760,CC.EXP
2269	006404			IF #CBIT SETIN CC.EXP OR #ZBIT SETIN CC.EXP THEN		
	006404	032767	000001 172110		BIT	#CBIT,CC.EXP
	006412	001004			BNE	\$50234
	006414	032767	000004 172100		BIT	#ZBIT,CC.EXP
	006422	001403			BEG	\$50235
	006424				\$50234:	
2270	006424			LET R0.EXP := #0		
	006424	005067	172074		CLR	R0.EXP
2271	006430			ELSE		
	006430	000404			BR	\$50236
	006432				\$50235:	
2272	006432			LET R0.EXP := R0 - R2		
	006432	010067	172066		MOV	R0,R0.EXP
	006436	160267	172062		SUB	R2,R0.EXP
2273	006442			ENDIF		
	006442				\$50236:	
2274	006442			LET R1.EXP := #0		
	006442	005067	172060		CLR	R1.EXP
2275	006446			LET R2.EXP := #0		
	006446	005067	172056		CLR	R2.EXP
2276	006452			LET R3.EXP := #0		
	006452	005067	172054		CLR	R3.EXP
2277	006456			LET R4.EXP := R4		
	006456	010467	172052		MOV	R4,R4.EXP
2278	006462			LET R5.EXP := R5		
	006462	010567	172050		MOV	R5,R5.EXP
2279	006466			LET BFADRS := DATA01		
	006466	016767	172072 172046		MOV	DATA01,BFADRS
2280	006474			IF DATA05 EQ #0 THEN		
	006474	005767	172074		TST	DATA05
	006500	001003			BNE	\$50237
2281	006502			CALL CLRBUF		
	006502	004767	047750		JSR	PC,CLRBUF
2282	006506			ELSE		
	006506	000411			BR	\$50240
	006510				\$50237:	
2283	006510			IF DATA05 EQ #177777 THEN		
	006510	026727	172060 177777		CMP	DATA05,#177777
	006516	001003			BNE	\$50241
2284	006520			CALL SETBUF		
	006520	004767	047774		JSR	PC,SETBUF
2285	006524			ELSE		
	006524	000402			BR	\$50242
	006526				\$50241:	
2286	006526			CALL ALTBUF		
	006526	004767	050032		JSR	PC,ALTBUF
2287	006532			ENDIF		

MOVTC

2288	006532				ENDIF	\$50242:	
2289	006532				LET BFADRS := DATA03	\$50240:	
2290	006532	016767	172032	172002	IF DATA06 EQ #0 THEN	MOV	DATA03,BFADRS
	006540	005767	172032			TST	DATA06
	006544	001003				BNE	\$50243
2291	006546				CALL CLRBUF	JSR	PC,CLRBUF
2292	006546	004767	047704		ELSE	BR	\$50244
	006552	000411				\$50243:	
2293	006554				IF DATA06 EQ #177777 THEN	CMP	DATA06,#177777
	006554	026727	172016	177777		BNE	\$50245
	006562	001003			CALL SETBUF	JSR	PC,SETBUF
2294	006564	004767	047730		ELSE	BR	\$50246
2295	006570	000402				\$50245:	
	006572				CALL ALTBUF	JSR	PC,ALTBUF
2296	006572	004767	047766		ENDIF	\$50246:	
2297	006576				ENDIF	\$50244:	
2298	006576				LET BF.EXP := @DATA03	MOV	@DATA03,BF.EXP
2299	006576	017767	171766	171734			
2300							
2301					;;		
2302					;; SET UP THE TRANSLATION TABLE		
2303					;-		
2304							
2305	006604				DECR TEMPO FROM #200 TO #1 BY #1	MOV	#200,TEMPO
	006604	012767	000200	171664		BR	\$50247
	006612	000402				\$50250:	
	006614					DEC	TEMPO
	006614	005367	171656			\$50247:	
	006620					CMP	TEMPO,#1
	006620	026727	171652	000001		BLT	\$50251
	006626	002402			LET (R5) := #0	CLR	(R5)
2306	006630				ENDDEC	BR	\$50250
	006630	005025				\$50251:	
2307	006632				LET R5 := #TRANS	MOV	#TRANS,R5
	006632	000770			LET (R5) :B= #63	MOVB	#63,(R5)
	006634				LET 252(R5) :B= #314	MOVB	#314,252(R5)
2308	006634	012705	063436		LET 377(R5) :B= #125	MOVB	#125,377(R5)
2309	006640						
	006640	112715	000063				
2310	006644						
	006644	112765	000314	000252			
2311	006652	112765	000125	000377			
	006652						
2312							
2313					;;		

MOVTC

```

2314 ; DO THE CIS INSTRUCTION
2315 ;
2316
2317 006660      ;
      006660 076032      ;
                                ;
2318                                ;
2319                                ;
2320                                ;
2321                                ;
2322                                ;
2323 006662      ;
      006662 016767 171110 171630      ;
2324 006670      ;
      006670 042767 177760 171622      ;
2325 006676      ;
      006676 026767 171616 171616      ;
      006704 001402      ;
2326 006706      ;
      006706 004767 050766      ;
2327 006712      ;
      006712      ;
2328 006712      ;
      006712 020067 171606      ;
      006716 001402      ;
2329 006720      ;
      006720 004767 050754      ;
2330 006724      ;
      006724      ;
2331 006724      ;
      006724 020167 171576      ;
      006730 001402      ;
2332 006732      ;
      006732 004767 050742      ;
2333 006736      ;
      006736      ;
2334 006736      ;
      006736 020267 171566      ;
      006742 001402      ;
2335 006744      ;
      006744 004767 050730      ;
2336 006750      ;
      006750      ;
2337 006750      ;
      006750 020367 171556      ;
      006754 001402      ;
2338 006756      ;
      006756 004767 050716      ;
2339 006762      ;
      006762      ;
2340 006762      ;
      006762 020467 171546      ;
      006766 001402      ;
2341 006770      ;
      006770 004767 050704      ;
2342 006774      ;
      006774      ;
2343 006774      ;

```

MOVTC

```

; DO THE CIS INSTRUCTION
;
;
;
;
; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS
;
;

```

```

LET CC := PSW
LET CC := CC CLR.BY #177760
IF CC NE CC.EXP THEN
    CALL ERROR
ENDIF
IF R0 NE R0.EXP THEN
    CALL ERROR
ENDIF
IF R1 NE R1.EXP THEN
    CALL ERROR
ENDIF
IF R2 NE R2.EXP THEN
    CALL ERROR
ENDIF
IF R3 NE R3.EXP THEN
    CALL ERROR
ENDIF
IF R4 NE R4.EXP THEN
    CALL ERROR
ENDIF
IF R5 NE R5.EXP THEN

```

```

MOV PSW,CC
BIC #177760,CC
CMP CC,CC.EXP
BEQ $50252
JSR PC,ERROR
$50252:
CMP R0,R0.EXP
BEQ $50253
JSR PC,ERROR
$50253:
CMP R1,R1.EXP
BEQ $50254
JSR PC,ERROR
$50254:
CMP R2,R2.EXP
BEQ $50255
JSR PC,ERROR
$50255:
CMP R3,R3.EXP
BEQ $50256
JSR PC,ERROR
$50256:
CMP R4,R4.EXP
BEQ $50257
JSR PC,ERROR
$50257:

```

MOVTC

006774	020567	171536							
2344 007000	001402								
	007002		CALL ERROR						
2345 007006	004767	050672							
	007006		ENDIF						
2346									
2347									
2348									
2349									
2350									
2351 007006			LET R0 := DATA03						
	007006	016700							
2352 007012			LET R2 :B= @DATA01						
	007012	117702							
2353 007016			LET R2 := R2 CLR.BY #177400						
	007016	042702							
2354 007022			LET R1 :B= TRANS(R2)						
	007022	116201							
2355 007026			LET ERRFLG := #0						
	007026	005067							
2356 007032			IF DATA00 HIS DATA02 THEN						
	007032	026767							
	007040	103416							
2357 007042			LET R3 := DATA02						
	007042	016703							
2358 007046			WHILE R3 NE #0 AND ERRFLG EQ #0 DO						
	007046								
	007046	005703							
	007050	001411							
	007052	005767							
	007056	001006							
2359 007060			IFB (R0). NE R1 THEN						
	007060	122001							
	007062	001402							
2360 007064			CALL ERROR						
	007064	004767							
2361 007070			ENDIF						
	007070								
2362 007070			LET R3 := R3 - #1						
	007070	005303							
2363 007072			ENDDO						
	007072	000765							
	007074								
2364 007074			ELSE						
	007074	000440							
	007076								
2365 007076			LET R3 := DATA00						
	007076	016703							
2366 007102			WHILE R3 NE #0 AND ERRFLG EQ #0 DO						
	007102								
	007102	005703							
	007104	001411							
	007106	005767							
	007112	001006							
2367 007114			IFB (R0). NE R1 THEN						
	007114	122001							

\$50260:

```

;*
; CHECK ENTIRE DESTINATION BUFFER FOR CORRECT RESULTS
;-

```

\$50262:

\$50264:

\$50263:

\$50261:

\$50266:

```

CMP      R5,R5,EXP
BEQ      $50260

JSR      PC,ERROR

MOV      DATA03,R0
MOVB     @DATA01,R2
BIC      #177400,R2
MOVB     TRANS(R2),R1
CLR      ERRFLG
CMP      DATA00,DATA02
BLO      $50261
MOV      DATA02,R3

TST      R3
BEQ      $50263
TST      ERRFLG
BNE      $50263

CMPB     (R0).,R1
BEQ      $50264

JSR      PC,ERROR

DEC      R3
BR       $50262

BR       $50262

BR       $50265

MOV      DATA00,R3

TST      R3
BEQ      $50267
TST      ERRFLG
BNE      $50267

CMPB     (R0).,R1

```



MOVTC							
2368	007116	001402					
	007120			CALL ERROR		BEQ	\$50270
	007120	004767	050554			JSR	PC.ERROR
2369	007124			ENDIF			
	007124					\$50270:	
2370	007124			LET R3 := R3 - #1			
	007124	005303				DEC	R3
2371	007126			ENDDO			
	007126	000765				BR	\$50266
	007130					\$50267:	
2372	007130			LET R0 := DATA03 * DATA00			
	007130	016700	171434			MOV	DATA03,R0
	007134	066700	171422			ADD	DATA00,R0
2373	007140			LET R3 := DATA02 - DATA00			
	007140	016703	171422			MOV	DATA02,R3
	007144	166703	171412			SUB	DATA00,R3
2374	007150			REPEAT			
	007150					\$50271:	
2375	007150			IFB (R0) * NE DATA04 THEN			
	007150	122067	171416			CMPB	(R0) *, DATA04
	007154	001402				BEQ	\$50272
2376	007156			CALL ERROR			
	007156	004767	050516			JSR	PC.ERROR
2377	007162			ENDIF			
	007162					\$50272:	
2378	007162			LET R3 := R3 - #1			
	007162	005303				DEC	R3
2379	007164			UNTIL R3 EQ #0 OR ERRFLG NE #0			
	007164	005703				TST	R3
	007166	001403				BEQ	\$50273
	007170	005767	171262			TST	ERRFLG
	007174	001765				BEQ	\$50271
	007176					\$50273:	
2380	007176			ENDIF			
	007176					\$50265:	
2381	007176			IF DATA02 LO #BUFLEN THEN			
	007176	026727	171364	000100		CMP	DATA02, #BUFLEN
	007204	103027				BHIS	\$50274
2382	007206			LET TEMPO := DATA03 * #BUFLEN - #1			
	007206	016767	171356	171262		MOV	DATA03, TEMPO
	007214	062767	000100	171254		ADD	#BUFLEN, TEMPO
	007222	005367	171250			DEC	TEMPO
2383	007226			LET R0 := DATA03 * DATA02			
	007226	016700	171336			MOV	DATA03, R0
	007232	066700	171330			ADD	DATA02, R0
2384	007236			REPEAT			
	007236					\$50275:	
2385	007236			IFB (R0) * NE BF.EXP THEN			
	007236	122067	171276			CMPB	(R0) *, BF.EXP
	007242	001402				BEQ	\$50276
2386	007244			CALL ERROR			
	007244	004767	050430			JSR	PC.ERROR
2387	007250			ENDIF			
	007250					\$50276:	
2388	007250			UNTIL R0 EQ TEMPO OR ERRFLG NE #0			
	007250	020067	171222			CMP	R0, TEMPO
	007254	001403				BEQ	\$50277

MOVTC

	007256	005767	171174				TST	ERRFLG
	007262	001765					BEQ	\$50275
	007264					\$50277:		
2389	007264			ENDIF				
	007264					\$50274:		
2390	007264			LET TSTCAS := TSTCAS + #16			ADD	#16, TSTCAS
	007264	062767	000016	171252	LET COUNT := COUNT - #1		DEC	COUNT
2391	007272							
	007272	005367	171172					
2392				:UNTIL COUNT EQ #0			TST	COUNT
2393	007276			IF COUNT NE #0 THEN			BEQ	\$50300
	007276	005767	171166					
	007302	001402						
2394	007304			INLINE <JMP \$5032>			JMP	\$5032
	007304	000167	176766					
2395	007310			ENDIF		\$50300:		
	007310							
2396								
2397								
2398								
2399				::				
2400				:: DO ONE INLINE CASE - MOVTCI				
2401				::-				
2402	007310			LET BFADRS := #BUFF1			MOV	#BUFF1, BFADRS
	007310	012767	063236	171224	CALL CLRBUF		JSR	PC, CLRBUF
2403	007316							
	007316	004767	047134		LET BFADRS := #BUFF2		MOV	#BUFF2, BFADRS
2404	007322							
	007322	012767	063336	171212	CALL SETBUF		JSR	PC, SETBUF
2405	007330							
	007330	004767	047164		LET R0 := #125252		MOV	#125252, R0
2406	007334				LET R1 := #125252		MOV	#125252, R1
	007334	012700	125252		LET R2 := #125252		MOV	#125252, R2
2407	007340				LET R3 := #125252		MOV	#125252, R3
	007340	012701	125252		LET R4 := #125252		MOV	#125252, R4
2408	007344				LET R5 := #125252		MOV	#125252, R5
	007344	012702	125252		LET TRANS := #70707		MOV	#70707, TRANS
2409	007350				INLINE <MOVTCI>		MOVTCI	
	007350	012703	125252		INLINE <.WORD TMVTCI>		.WORD	TMVTCI
2410	007354				INLINE <.WORD TMVTCI+4>		.WORD	TMVTCI+4
	007354	012704	125252		INLINE <.WORD 40>		.WORD	40
2411	007360				INLINE <.WORD TRANS>		.WORD	TRANS
	007360	012705	125252		IFCOND CS THEN		BCC	\$50301
2412	007364							
	007364	012767	070707	054044				
2413	007372							
	007372	076132						
2414	007374							
	007374	007700						
2415	007376							
	007376	007704						
2416	007400							
	007400	000040						
2417	007402							
	007402	063436						
2418	007404							
	007404	103003						

MOVTC

2419	007406			IFCOND EQ THEN		
	007406	001002			BNE	\$50302
2420	007410			CALL ERROR		
	007410	004767	050264		JSR	PC.ERROR
2421	007414			ENDIF		
	007414					\$50302:
2422	007414			ENDIF		
	007414					\$50301:
2423	007414			IF R0 NE #125252 THEN		
	007414	020027	125252		CMP	R0,#125252
	007420	001402			BEQ	\$50303
2424	007422			CALL ERROR		
	007422	004767	050252		JSR	PC.ERROR
2425	007426			ENDIF		
	007426					\$50303:
2426	007426			IF R1 NE #125252 THEN		
	007426	020127	125252		CMP	R1,#125252
	007432	001402			BEQ	\$50304
2427	007434			CALL ERROR		
	007434	004767	050240		JSR	PC.ERROR
2428	007440			ENDIF		
	007440					\$50304:
2429	007440			IF R2 NE #125252 THEN		
	007440	020227	125252		CMP	R2,#125252
	007444	001402			BEQ	\$50305
2430	007446			CALL ERROR		
	007446	004767	050226		JSR	PC.ERROR
2431	007452			ENDIF		
	007452					\$50305:
2432	007452			IF R3 NE #125252 THEN		
	007452	020327	125252		CMP	R3,#125252
	007456	001402			BEQ	\$50306
2433	007460			CALL ERROR		
	007460	004767	050214		JSR	PC.ERROR
2434	007464			ENDIF		
	007464					\$50306:
2435	007464			IF R4 NE #125252 THEN		
	007464	020427	125252		CMP	R4,#125252
	007470	001402			BEQ	\$50307
2436	007472			CALL ERROR		
	007472	004767	050202		JSR	PC.ERROR
2437	007476			ENDIF		
	007476					\$50307:
2438	007476			IF R5 NE #125252 THEN		
	007476	020527	125252		CMP	R5,#125252
	007502	001402			BEQ	\$50310
2439	007504			CALL ERROR		
	007504	004767	050170		JSR	PC.ERROR
2440	007510			ENDIF		
	007510					\$50310:
2441	007510			IF BUFF2-NE #143707 THEN		
	007510	026727	053622 143707		CMP	BUFF2,#143707
	007516	001402			BEQ	\$50311
2442	007520			CALL ERROR		
	007520	004767	050154		JSR	PC.ERROR
2443	007524			ENDIF		
	007524					\$50311:

MOVTC

```

2444 007524          IF BUFF2+2 NE #20040 THEN
      007524 026727 053610 020040          CMP      BUFF2+2,#20040
      007532 001402          BEQ      $50312
2445 007534          CALL ERROR
      007534 004767 050140          JSR      PC,ERROR
2446 007540          ENDIF
      007540          $50312:
2447 007540          LET RO := #BUFF2 + #4
      007540 012700 063336          MOV      #BUFF2,RO
      007544 062700 000004          ADD      #4,RO
2448 007550          LET ERRFLG := #0
      007550 005067 170702          CLR      ERRFLG
2449 007554          REPEAT
      007554          $50313:
2450 007554          IF (RO)+ NE #177777 THEN
      007554 022027 177777          CMP      (RO)+,#177777
      007560 001402          BEQ      $50314
2451 007562          CALL ERROR
      007562 004767 050112          JSR      PC,ERROR
2452 007566          ENDIF
      007566          $50314:
2453 007566          UNTIL RO EQ #BUFF2+BUFLN OR ERRFLG NE #0
      007566 020027 063436          CMP      RO,#BUFF2+BUFLN
      007572 001403          BEQ      $50315
      007574 005767 170656          TST      ERRFLG
      007600 001765          BEQ      $50313
      007602          $50315:
2454 007602          INLINE <JMP XLOCC>
      007602 000167 000102          JMP      XLOCC

```

```

2455
2456 ;*
2457 ; TABLE OF TEST CASES FOR MOVTC
2458 ;-

```

2460	007606	000004	TMOVTC: .WORD	4		;#OF TEST CASES FOR MOVTC
2461						
2462	007610	000061	.WORD	61		;SRC.LEN
2463	007612	063336	.WORD	BUFF2		;SRC.ADR
2464	007614	000027	.WORD	27		;DST.LEN
2465	007616	063236	.WORD	BUFF1		;DST.ADR
2466	007620	000040	.WORD	40		;FILL CHARACTER
2467	007622	000000	.WORD	0		;SRC.BUFF.STATE
2468	007624	000001	.WORD	1		;DST.BUFF.STATE
2469						
2470	007626	000055	.WORD	55		;TEST CASE NUMBER 2
2471	007630	063336	.WORD	BUFF2		
2472	007632	000075	.WORD	75		
2473	007634	063236	.WORD	BUFF1		
2474	007636	000100	.WORD	100		
2475	007640	000001	.WORD	1		
2476	007642	000000	.WORD	0		
2477						
2478	007644	000032	.WORD	32		;TEST CASE NUMBER 3
2479	007646	063236	.WORD	BUFF1		
2480	007650	000031	.WORD	31		
2481	007652	063336	.WORD	BUFF2		
2482	007654	000073	.WORD	73		

MOVTC

2483	007656	000000	.WORD	0	
2484	007660	177777	.WORD	-1	
2485					
2486	007662	000005	.WORD	5	;TEST CASE NUMBER 4
2487	007664	063236	.WORD	BUFF1	
2488	007666	000026	.WORD	26	
2489	007670	063336	.WORD	BUFF2	
2490	007672	000020	.WORD	20	
2491	007674	177777	.WORD	-1	
2492	007676	000001	.WORD	1	
2493					
2494	007700	000002	TMVTCI: .WORD	2	;TEST CASE FOR INLINE
2495	007702	063236	.WORD	BUFF1	
2496	007704	000004	.WORD	4	
2497	007706	063336	.WORD	BUFF2	
2498					

LOCC

```

2500 .SBTTL LOCC
2501 ;*****
2502 ;
2503 ; LOCC TEST
2504 ;
2505 ;*****
2506
2507 007710 ROUTINE XLOCC
      007710
2508 007710          XLOCC:
      007710 000004          SCOPE
2509 007712          SCOPE
      007712 000240          NOP
2510 007714          LET $TIMES := #20          ;ITERATION COUNT
      007714 012767 000020 170462          ;MOV #20,$TIMES
2511 007722          LET COUNT := TLOCC          ;GET TEST COUNT
      007722 016767 000616 170540          ;MOV TLOCC,COUNT
2512 007730          LET TSTCAS := #TLOCC*2          ;POINT TO FIRST TEST CASE
      007730 012767 010546 170606          ;MOV #TLOCC-2,TSTCAS
2513 ;REPEAT
2514 007736          ;REPEAT
      007736          ;   INLINE <$5040:>
2515 ;
2516 ;*
2517 ; GET TEST DATA FROM TABLE
2518 ;
2519 ;
2520 007736          LET R0 := TSTCAS
      007736 016700 170602          ;MOV TSTCAS,R0
2521 007742          LET DATA0 := (R0)*          ;SRC.LEN
      007742 012067 170614          ;MOV (R0),DATA0
2522 007746          LET DATA01 := (R0)*          ;CHAR.SEARCHED FOR
      007746 012067 170612          ;MOV (R0),DATA01
2523 007752          LET DATA02 := (R0)          ;OFFSET
      007752 011067 170610          ;MOV (R0),DATA02
2524 ;
2525 ;*
2526 ; SET UP GPR'S,
2527 ; EXPECTED GPR'S AND CONDITION CODES,
2528 ; AND SOURCE BUFFER.
2529 ;-
2530 ;
2531 007756          LET R0 := DATA00
      007756 016700 170600          ;MOV DATA00,R0
2532 007762          LET R1 := #BUFF1
      007762 012701 063236          ;MOV #BUFF1,R1
2533 007766          LET R4 := DATA01
      007766 016704 170572          ;MOV DATA01,R4
2534 007772          INLINE <BIT #BIT00,COUNT>
      007772 032767 000001 170470          ;BIT #BIT00,COUNT
2535 010000          IFCOND EQ THEN
      010000 001004          ;BNE $50320
2536 010002          LET R2 := #0
      010002 005002          ;CLR R2
2537 010004          LET R3 := #0
      010004 005003          ;CLR R3
2538 010006          LET R5 := #0

```

LOCC

```

010006 005005
2539 010010
010010 000406
010012
2540 010012
010012 012702 177777
2541 010016
010016 012703 177777
2542 010022
010022 012705 177777
2543 010026
010026
2544 010026
010026 026767 170534 170526
010034 101411
2545 010036
010036 005067 170462
2546 010042
010042 012767 063236 170456
010050 066767 170506 170450
2547 010056
010056 000414
010060
2548 010060
010060 016767 170476 170436
010066 166767 170474 170430
2549 010074
010074 012767 063236 170424
010102 066767 170460 170416
2550 010110
010110
2551 010110
010110 010267 170414
2552 010114
010114 010367 170412
2553 010120
010120 010467 170410
2554 010124
010124 010567 170406
2555 010130
010130 005767 170370
2556 010134
010134 016767 167636 170360
2557 010142
010142 042767 177760 170352
2558 010150
010150 012767 063236 170364
2559 010156
010156 004767 046274
2560 010162
010162 012767 063236 170306
010170 066767 170372 170300
2561 010176
010176 116777 170362 170272

```

```

ELSE
LET R2 := #177777
LET R3 := #177777
LET R5 := #177777
ENDIF
IF DATA02 HI DATA00 THEN
LET R0.EXP := #0
LET R1.EXP := #BUFF1 + DATA00
ELSE
LET R0.EXP := DATA00 - DATA02
LET R1.EXP := #BUFF1 + DATA02
ENDIF
LET R2.EXP := R2
LET R3.EXP := R3
LET R4.EXP := R4
LET R5.EXP := R5
INLINE <TST R0.EXP>
LET CC.EXP := PSW
LET CC.EXP := CC.EXP CLR.BY #177760
LET BFADRS := #BUFF1
CALL CLRBUF
LET TEMPO := #BUFF1 + DATA02
LET @TEMPO :B= DATA01

```

```

CLR R5
BR $50321
$50320:
MOV #177777,R2
MOV #177777,R3
MOV #177777,R5
$50321:
CMP DATA02,DATA00
BLOS $50322
CLR R0.EXP
MOV #BUFF1,R1.EXP
ADD DATA00,R1.EXP
BR $50323
$50322:
MOV DATA00,R0.EXP
SUB DATA02,R0.EXP
MOV #BUFF1,R1.EXP
ADD DATA02,R1.EXP
$50323:
MOV R2,R2.EXP
MOV R3,R3.EXP
MOV R4,R4.EXP
MOV R5,R5.EXP
TST R0.EXP
MOV PSW,CC.EXP
BIC #177760,CC.EXP
MOV #BUFF1,BFADRS
JSR PC,CLRBUF
MOV #BUFF1,TEMPO
ADD DATA02,TEMPO
MOVB DATA01,@TEMPO

```

2562  
2563  
2564

::  
: DO THE CIS INSTRUCTION.

LOCC

```

2565 ;
2566 ;
2567 010204      ;      INLINE <LOCC>
      010204 076040 ;
2568 ;
2569 ;
2570 ; *
2571 ; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS
2572 ;
2573 010206      ;      LET CC := PSW
      010206 016767 167564 170304 ;
2574 010214      ;      LET CC := CC CLR.BY #177760
      010214 042767 177760 170276 ;
2575 010222      ;      IF CC NE CC.EXP THEN
      010222 026767 170272 170272 ;
      010230 001402 ;
2576 010232      ;      CALL ERROR
      010232 004767 047442 ;
2577 010236      ;      ENDIF
      010236 ;
2578 010236      ;      IF R0 NE R0.EXP THEN
      010236 020067 170262 ;
      010242 001402 ;
2579 010244      ;      CALL ERROR
      010244 004767 047430 ;
2580 010250      ;      ENDIF
      010250 ;
2581 010250      ;      IF R1 NE R1.EXP THEN
      010250 020167 170252 ;
      010254 001402 ;
2582 010256      ;      CALL ERROR
      010256 004767 047416 ;
2583 010262      ;      ENDIF
      010262 ;
2584 010262      ;      IF R2 NE R2.EXP THEN
      010262 020267 170242 ;
      010266 001402 ;
2585 010270      ;      CALL ERROR
      010270 004767 047404 ;
2586 010274      ;      ENDIF
      010274 ;
2587 010274      ;      IF R3 NE R3.EXP THEN
      010274 020367 170232 ;
      010300 001402 ;
2588 010302      ;      CALL ERROR
      010302 004767 047372 ;
2589 010306      ;      ENDIF
      010306 ;
2590 010306      ;      IF R4 NE R4.EXP THEN
      010306 020467 170222 ;
      010312 001402 ;
2591 010314      ;      CALL ERROR
      010314 004767 047360 ;
2592 010320      ;      ENDIF
      010320 ;
2593 010320      ;      IF R5 NE R5.EXP THEN
      010320 020567 170212 ;

```

LOCC

```

MOV PSW,CC
BIC #177760,CC
CMP CC,CC.EXP
BEQ $50324
JSR PC,ERROR
$50324:
CMP R0,R0.EXP
BEQ $50325
JSR PC,ERROR
$50325:
CMP R1,R1.EXP
BEQ $50326
JSR PC,ERROR
$50326:
CMP R2,R2.EXP
BEQ $50327
JSR PC,ERROR
$50327:
CMP R3,R3.EXP
BEQ $50330
JSR PC,ERROR
$50330:
CMP R4,R4.EXP
BEQ $50331
JSR PC,ERROR
$50331:
CMP R5,R5.EXP

```



LOCC

2594	010324	001402			CALL ERROR	BEQ	\$50332
	010326					JSR	PC.ERROR
2595	010332	004767	047346		ENDIF		
	010332					\$50332:	
2596	010332				LET TSTCAS := TSTCAS + #6	ADD	#6.TSTCAS
	010332	062767	000006	170204		DEC	COUNT
2597	010340				LET COUNT := COUNT - #1		
	010340	005367	170124			TST	COUNT
2598					;UNTIL COUNT EQ #0	BEQ	\$50333
2599	010344				IF COUNT NE #0 THEN		
	010344	005767	170120			JMP	\$5040
	010350	001402					
2600	010352				INLINE <JMP \$5040>		
	010352	000167	177360			\$50333:	
2601	010356				ENDIF		
	010356						
2602							
2603							
2604							
2605					;+ DO ONE INLINE CASE - LOCCI		
2606					;-		
2607							
2608	010356				LET BFADRS := #BUFF1		
	010356	012767	063236	170156		MOV	#BUFF1,BFADRS
2609	010364				CALL SETBUF	JSR	PC.SETBUF
	010364	004767	046130			JSR	PC.CLRREG
2610	010370				CALL CLRREG		
	010370	004767	046044			MOV	#BUFF1,TEMPO
2611	010374				LET TEMPO := #BUFF1 + #25	ADD	#25,TEMPO
	010374	012767	063236	170074		MOVB	#40,@TEMPO
	010402	062767	000025	170066		LOCCI	
2612	010410				LET @TEMPO :B= #40	.WORD	TLOCCI
	010410	112777	000040	170060		.WORD	40
2613	010416				INLINE <LOCCI>	MOV	PSW,TEMPO
	010416	076140				BIT	#17,TEMPO
2614	010420				INLINE <.WORD TLOCCI>	BEQ	\$50334
	010420	010570				JSR	PC.ERROR
2615	010422				INLINE <.WORD 40>		
	010422	000040				\$50334:	
2616	010424				LET TEMPO := PSW	MOV	#BUFLEN 25,TEMPO
	010424	016767	167346	170044		CMP	RO,TEMPO
2617	010432				INLINE <BIT #17,TEMPO>	BEQ	\$50335
	010432	032767	000017	170036		JSR	PC.ERROR
2618	010440				IFCOND NE THEN		
	010440	001402					
2619	010442				CALL ERROR		
	010442	004767	047232				
2620	010446				ENDIF		
	010446						
2621	010446				LET TEMPO := #BUFLEN-25		
	010446	012767	000053	170022			
2622	010454				IF RO NE TEMPO THEN		
	010454	020067	170016				
	010460	001402					
2623	010462				CALL ERROR		
	010462	004767	047212				

LOCC

```

2624 010466          ENDF
      010466
2625 010466          IF R1 NE #BUFF1.25 THEN
      010466 020127 063263
      010472 001402
2626 010474          CALL ERROR
      010474 004767 047200
2627 010500          ENDF
      010500
2628 010500          IF R2 NE #0 THEN
      010500 005702
      010502 001402
2629 010504          CALL ERROR
      010504 004767 047170
2630 010510          ENDF
      010510
2631 010510          IF R3 NE #0 THEN
      010510 005703
      010512 001402
2632 010514          CALL ERROR
      010514 004767 047160
2633 010520          ENDF
      010520
2634 010520          IF R4 NE #0 THEN
      010520 005704
      010522 001402
2635 010524          CALL ERROR
      010524 004767 047150
2636 010530          ENDF
      010530
2637 010530          IF R5 NE #0 THEN
      010530 005705
      010532 001402
2638 010534          CALL ERROR
      010534 004767 047140
2639 010540          ENDF
      010540
2640 010540          INLINE <JMP XSKPC>
      010540 000167 000030

```

```

$50335:
      CMP      R1,#BUFF1.25
      BEQ      $50336
      JSR      PC,ERROR
$50336:
      TST      R2
      BEQ      $50337
      JSR      PC,ERROR
$50337:
      TST      R3
      BEQ      $50340
      JSR      PC,ERROR
$50340:
      TST      R4
      BEQ      $50341
      JSR      PC,ERROR
$50341:
      TST      R5
      BEQ      $50342
      JSR      PC,ERROR
$50342:
      JMP      XSKPC

```

```

2641
2642
2643 ;+
2644 ; TABLE OF TEST CASES FOR LOCC.
2645 ; -
2646 TLOCC: .WORD 3
2647
2648 .WORD 0
2649 .WORD 101
2650 .WORD 41
2651
2652 .WORD 60
2653 .WORD 40
2654 .WORD 26
2655
2656 .WORD 34
2657 .WORD 77
2658 .WORD 40

```

```

; # OF TEST CASES FOR LOCC
; SRC.LEN
; CHAR. SEARCHED FOR
; OFFSET
; TEST CASE NUMBER 2
; TEST CASE NUMBER 3

```

LOCC

2659

2660

2661 010570 000100

2662 010572 063236

2663

TLOCCI: .WORD BUFLN  
.WORD BUFF1

;TEST CASE FOR INLINE

SKPC

```

2665 .SBTTL SKPC
2666 ;*****
2667 ;
2668 ; SKPC TEST
2669 ;
2670 ;*****
2671 ;
2672 010574 ROUTINE XSKPC
      010574
2673 010574          XSKPC:
      010574 000004          SCOPE
2674 010576          SCOPE
      010576 000240          NOP
2675 010600          LET $TIMES := #20          ;ITERATION COUNT
      010600 012767 000020 167576          ;MOV #20,$TIMES
2676 010606          LET COUNT := TSKPC          ;GET TEST COUNT
      010606 016767 000624 167654          ;MOV TSKPC,COUNT
2677 010614          LET TSTCAS := #TSKPC+2          ;POINT TO FIRST TEST CASE
      010614 012767 011440 167722          ;MOV #TSKPC+2,TSTCAS
2678 ;REPEAT
2679 010622          ;
      010622          ;
2680 ;
2681 ;+
2682 ; GET TEST DATA FROM TABLE
2683 ;-
2684 ;
2685 010622          LET R0 := TSTCAS
      010622 016700 167716          ;MOV TSTCAS,R0
2686 010626          LET DATA00 := (R0)+          ;SRC.LEN
      010626 012067 167730          ;MOV (R0)+,DATA00
2687 010632          LET DATA01 := (R0)+          ;NON-SEARCH CHAR
      010632 012067 167726          ;MOV (R0)+,DATA01
2688 010636          LET DATA02 := (R0)          ;OFFSET
      010636 011067 167724          ;MOV (R0),DATA02
2689 ;
2690 ;+
2691 ; SET UP GPR'S
2692 ; EXPECTED GPR'S AND CONDITION CODES,
2693 ; AND SOURCE BUFFER.
2694 ;-
2695 ;
2696 010642          LET R0 := DATA00
      010642 016700 167714          ;MOV DATA00,R0
2697 010646          LET R1 := #BUFF1
      010646 012701 063236          ;MOV #BUFF1,R1
2698 010652          LET R4 := #377
      010652 012704 000377          ;MOV #377,R4
2699 010656          INLINE <BIT #BIT00,COUNT>
      010656 032767 000001 167604          ;BIT #BIT00,COUNT
2700 010664          IFCOND EQ THEN
      010664 001004          ;BNE $50345
2701 010666          LET R2 := #0
      010666 005002          ;CLR R2
2702 010670          LET R3 := #0
      010670 005003          ;CLR R3
2703 010672          LET R5 := #0

```

SKPC

2704	010672	005005			ELSE	CLR	R5
	010674					BR	\$50346
	010676	000406				\$50345.	
2705	010676				LET R2 := #177777	MOV	#177777,R2
	010676	012702	177777			MOV	#177777,R3
2706	010702				LET R3 := #177777	MOV	#177777,R5
	010702	012703	177777				
2707	010706				LET R5 := #177777		
	010706	012705	177777				
2708	010712				ENDIF		
	010712					\$50346:	
2709	010712				IF DATA02 HI DATA00 THEN		
	010712	026767	167650	167642		CMP	DATA02,DATA00
	010720	101411				BLOS	\$50347
2710	010722				LET R0.EXP := #0	CLR	R0.EXP
	010722	005067	167576			MOV	#BUFF1,R1.EXP
2711	010726				LET R1.EXP := #BUFF1 + DATA00	ADD	DATA00,R1.EXP
	010726	012767	063236	167572			
	010734	066767	167622	167564			
2712	010742				ELSE	BR	\$50350
	010742	000414				\$50347:	
	010744					MOV	DATA00,R0.EXP
2713	010744				LET R0.EXP := DATA00 - DATA02	SUB	DATA02,R0.EXP
	010744	016767	167612	167552			
	010752	166767	167610	167544			
2714	010760				LET R1.EXP := #BUFF1 + DATA02	MOV	#BUFF1,R1.EXP
	010760	012767	063236	167540		ADD	DATA02,R1.EXP
	010766	066767	167574	167532			
2715	010774				ENDIF		
	010774					\$50350:	
2716	010774				LET R2.EXP := R2	MOV	R2,R2.EXP
	010774	010267	167530				
2717	011000				LET R3.EXP := R3	MOV	R3,R3.EXP
	011000	010367	167526				
2718	011004				LET R4.EXP := R4	MOV	R4,R4.EXP
	011004	010467	167524				
2719	011010				LET R5.EXP := R5	MOV	R5,R5.EXP
	011010	010567	167522				
2720	011014				INLINE <TST R0.EXP>	TST	R0.EXP
	011014	005767	167504				
2721	011020				LET CC.EXP := PSW	MOV	PSW,CC.EXP
	011020	016767	166752	167474			
2722	011026				LET CC.EXP := CC.EXP CLR.BY #177760	BIC	#177760,CC.EXP
	011026	042767	177760	167466			
2723	011034				LET BFADRS := #BUFF1	MOV	#BUFF1,BFADRS
	011034	012767	063236	167500			
2724	011042				CALL SETBUF ;FILL BUFFER WITH SEARCH CHAR	JSR	PC,SETBUF
	011042	004767	045452				
2725	011046				LET TEMPO := #BUFF1 + DATA02	MOV	#BUFF1,TEMPO
	011046	012767	063236	167422		ADD	DATA02,TEMPO
	011054	066767	167506	167414			
2726	011062				LET @TEMPO := DATA01	MOVB	DATA01,@TEMPO
	011062	116777	167476	167406			
2727							
2728							
2729					; DO THE CIS INSTRUCTION		

SKPC

```

2730 ;-
2731
2732 011070          ;-      INLINE <SKPC>
      011070 076041
2733
2734 ;+
2735 ; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS
2736 ;-
2737
2738 011072          LET CC := PSW
      011072 016767 166700 167420          MOV      PSW,CC
2739 011100          LET CC := CC CLR.BY #177760
      011100 042767 177760 167412          BIC      #177760,CC
2740 011106          IF CC NE CC.EXP THEN
      011106 026767 167406 167406          CMP      CC,CC.EXP
      011114 001402          BEQ      $50351
2741 011116          CALL ERROR
      011116 004767 046556          JSR      PC,ERROR
2742 011122          ENDIF
2743 011122          IF R0 NE R0.EXP THEN          $50351:
      011122 020067 167376          CMP      R0,R0.EXP
      011126 001402          BEQ      $50352
2744 011130          CALL ERROR
      011130 004767 046544          JSR      PC,ERROR
2745 011134          ENDIF
2746 011134          IF R1 NE R1.EXP THEN          $50352:
      011134 020167 167366          CMP      R1,R1.EXP
      011140 001402          BEQ      $50353
2747 011142          CALL ERROR
      011142 004767 046532          JSR      PC,ERROR
2748 011146          ENDIF
2749 011146          IF R2 NE R2.EXP THEN          $50353:
      011146 020267 167356          CMP      R2,R2.EXP
      011152 001402          BEQ      $50354
2750 011154          CALL ERROR
      011154 004767 046520          JSR      PC,ERROR
2751 011160          ENDIF
2752 011160          IF R3 NE R3.EXP THEN          $50354:
      011160 020367 167346          CMP      R3,R3.EXP
      011164 001402          BEQ      $50355
2753 011166          CALL ERROR
      011166 004767 046506          JSR      PC,ERROR
2754 011172          ENDIF
2755 011172          IF R4 NE R4.EXP THEN          $50355:
      011172 020467 167336          CMP      R4,R4.EXP
      011176 001402          BEQ      $50356
2756 011200          CALL ERROR
      011200 004767 046474          JSR      PC,ERROR
2757 011204          ENDIF
2758 011204          IF R5 NE R5.EXP THEN          $50356:
      011204 020567 167326          CMP      R5,R5.EXP

```

```

SKPC
2759 011210 001402          CALL ERROR          BEQ    $50357
      011212          CALL ERROR          JSR    PC,ERROR
2760 011212 004767 046462  ENDIF
      011216          ENDIF          $50357:
2761 011216 062767 000006 167320 LET TSTCAS := TSTCAS + #6 ADD    #6,TSTCAS
      011216 062767 000006 167320 LET COUNT := COUNT - #1 DEC    COUNT
2762 011224 005367 167240 ;UNTIL COUNT EQ #0
      011224 005367 167240 IF COUNT NE #0 THEN TST    COUNT
2763 011230 005767 167234          BEQ    $50360
2764 011230 001402          TST    COUNT
      011234 001402          BEQ    $50360
2765 011236 000167 177360          JMP    $5041
      011236 000167 177360          JMP    $5041
2766 011242          ENDIF          $50360:
      011242          ENDIF
2767
2768
2769
2770 ;*
2771 ;DO ONE INLINE CASE - SKPCI
2772 ;-
2773 011242          LET BFADRS := #BUFF1
      011242 012767 063236 167272 MOV    #BUFF1,BFADRS
2774 011250          CALL SETBUF          JSR    PC,SETBUF
      011250 004767 045244          JSR    PC,SETBUF
2775 011254          CALL CLRREG          JSR    PC,CLRREG
      011254 004767 045160          JSR    PC,CLRREG
2776 011260          LET TEMPO := #BUFF1 + #25 MOV    #BUFF1,TEMPO
      011260 012767 063236 167210 ADD    #25,TEMPO
      011266 062767 000025 167202
2777 011274          LET @TEMPO :B= #40 MOVB   #40,@TEMPO
      011274 112777 000040 167174
2778 011302          INLINE <SKPCI>          SKPCI
      011302 076141
2779 011304          INLINE <.WORD TSKPCI> .WORD  TSKPCI
      011304 011454          .WORD  TSKPCI
2780 011306          INLINE <.WORD 377> .WORD  377
      011306 000377          .WORD  377
2781 011310          LET TEMPO := PSW MOV    PSW,TEMPO
      011310 016767 166462 167160
2782 011316          INLINE <BIT #17,TEMPO> BIT    #17,TEMPO
      011316 032767 000017 167152
2783 011324          IFCOND NE THEN BEQ    $50361
      011324 001402          BEQ    $50361
2784 011326          CALL ERROR          JSR    PC,ERROR
      011326 004767 046346          JSR    PC,ERROR
2785 011332          ENDIF
      011332          ENDIF          $50361:
2786 011332          LET TEMPO := #BUFLEN + #25 MOV    #BUFLEN,TEMPO
      011332 012767 000100 167136 SUB    #25,TEMPO
      011340 162767 000025 167130
2787 011346          IF RO NE TEMPO THEN CMP    RO,TEMPO
      011346 020067 167124          BEQ    $50362
      011352 001402          BEQ    $50362
2788 011354          CALL ERROR

```

SKPC

```

2789 011354 004767 046320          JSR      PC,ERROR
      011360          ENDIF          $50362:
2790 011360          IF R1 NE #BUFF1+25 THEN
      011360 020127 063263          CMP      R1,#BUFF1+25
      011364 001402          BEQ      $50363
2791 011366          CALL ERROR
      011366 004767 046306          JSR      PC,ERROR
2792 011372          ENDIF
      011372          $50363:
2793 011372          IF R2 NE #0 THEN
      011372 005702          TST      R2
      011374 001402          BEQ      $50364
2794 011376          CALL ERROR
      011376 004767 046276          JSR      PC,ERROR
2795 011402          ENDIF
      011402          $50364:
2796 011402          IF R3 NE #0 THEN
      011402 005703          TST      R3
      011404 001402          BEQ      $50365
2797 011406          CALL ERROR
      011406 004767 046266          JSR      PC,ERROR
2798 011412          ENDIF
      011412          $50365:
2799 011412          IF R4 NE #0 THEN
      011412 005704          TST      R4
      011414 001402          BEQ      $50366
2800 011416          CALL ERROR
      011416 004767 046256          JSR      PC,ERROR
2801 011422          ENDIF
      011422          $50366:
2802 011422          IF R5 NE #0 THEN
      011422 005705          TST      R5
      011424 001402          BEQ      $50367
2803 011426          CALL ERROR
      011426 004767 046246          JSR      PC,ERROR
2804 011432          ENDIF
      011432          $50367:
2805 011432          INLINE <JMP      XSCANC>
      011432 000167 000022          JMP      XSCANC

```

```

2806
2807      ;+
2808      ; TABLE OF TEST CASES FOR SKPC.
2809      ;-
2810
2811 011436 000002      TSKPC:  .WORD  2          ;# OF TEST CASES FOR SKPC
2812
2813 011440 000040          .WORD  40          ;SRC.LEN
2814 011442 000100          .WORD 100          ;NON-SEARCH CHAR.
2815 011444 000060          .WORD  60          ;OFFSET
2816
2817 011446 000050          .WORD  50          ;TEST CASE NUMBER 2
2818 011450 000077          .WORD  77
2819 011452 000030          .WORD  30
2820
2821
2822 011454 000100      TSKPCI:  .WORD  BUFLen      ;TEST CASE FOR INLINE

```



SKPC

2823 011456 063236  
2824

.WORD BUFF1

SCANC

```

2826 .SBTTL SCANC
2827 ;:*****
2828 ;
2829 ; SCANC TEST
2830 ;
2831 ;:*****
2832
2833 011460 ROUTINE XSCANC
2834 011460 XSCANC:
2835 011460 000004 INLINE <SCOPE> SCOPE
2836 011462 000240 INLINE <NOP> NOP
2837 011464 012767 000020 166712 LET $TIMES := #20 ;ITERATION COUNT
2838 011472 016767 000710 166770 LET COUNT := TSCANC ;GET TEST COUNT
2839 011500 012767 012410 167036 LET TSTCAS := #TSCANC*2 ;POINT TO FIRST TEST CASE
2840 011506 ;REPEAT
2841 011506 ;REPEAT
2842 ;
2843 ; GET TEST DATA FROM TABLE
2844 ;
2845
2846 011506 LET R0 := TSTCAS
2847 011512 016700 167032 MOV TSTCAS,R0
2848 011516 012067 167044 LET DATA00 := (R0) ;SRC.LEN
2849 011516 011067 167042 LET DATA01 := (R0) ;OFFSET
2850 ;
2851 ; SET UP GPR'S,
2852 ; EXPECTED GPR'S AND CONDITION CODES
2853 ; SOURCE BUFFER, AND CHARACTER SET TABLE.
2854 ;
2855
2856 011522 LET R0 := DATA00
2857 011526 016700 167034 MOV DATA00,R0
2858 011532 012701 063236 LET R1 := #BUFF1
2859 011536 012704 000001 LET R4 := #1 ;MASK
2860 011542 012705 064036 LET R5 := #CHRSET
2861 011550 032767 000001 166720 INLINE <BIT #BIT00,COUNT>
2862 011552 001003 IFCOND EQ THEN
2863 011554 005002 LET R2 := #0
2864 011556 005003 LET R3 := #0
ELSE

```



SCANC

011756	000773								
2889	011760				LET R5 := #CHRSET				
	011760								
2890	011760	012705	064036						
	011764				LET TEMPO := #CHRSET * #125				
	011764	012767	064036	166504					
	011772	062767	000125	166476					
2891	012000				LET @TEMPO :B= #1				
	012000	112777	000001	166470					
2892									
2893									
2894					:*				
2895					: DO THE CIS INSTRUCTION				
2896					:				
2897	012006				INLINE <SCANC>				
	012006	076042							
2898									
2899									
2900					:*				
2901					: CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS				
2902					:-				
2903	012010				LET CC := PSW				
	012010	016767	165762	166502					
2904	012016				LET CC := CC CLR.BY #177760				
	012016	042767	177760	166474					
2905	012024				IF CC NE CC.EXP THEN				
	012024	026767	166470	166470					
	012032	001402							
2906	012034				CALL ERROR				
	012034	004767	045640						
2907	012040				ENDIF				
	012040								
2908	012040				IF R0 NE R0.EXP THEN				
	012040	020067	166460						
	012044	001402							
2909	012046				CALL ERROR				
	012046	004767	045626						
2910	012052				ENDIF				
	012052								
2911	012052				IF R1 NE R1.EXP THEN				
	012052	020167	166450						
	012056	001402							
2912	012060				CALL ERROR				
	012060	004767	045614						
2913	012064				ENDIF				
	012064								
2914	012064				IF R2 NE R2.EXP THEN				
	012064	020267	166440						
	012070	001402							
2915	012072				CALL ERROR				
	012072	004767	045602						
2916	012076				ENDIF				
	012076								
2917	012076				IF R3 NE R3.EXP THEN				
	012076	020367	166430						
	012102	001402							
2918	012104				CALL ERROR				

```

BR $50376
$50377:
;RESTORE R5
MOV #CHRSET,R5
MOV #CHRSET,TEMPO
ADD #125,TEMPO
;SET MASK IN CHAR SET TABLE
MOV #1,@TEMPO
SCANC
MOV PSW,CC
BIC #177760,CC
CMP CC,CC.EXP
BEQ $50400
JSR PC,ERROR
$50400:
CMP R0,R0.EXP
BEQ $50401
JSR PC,ERROR
$50401:
CMP R1,R1.EXP
BEQ $50402
JSR PC,ERROR
$50402:
CMP R2,R2.EXP
BEQ $50403
JSR PC,ERROR
$50403:
CMP R3,R3.EXP
BEQ $50404

```

```

SCANC
2919 012104 004767 045570          JSR      PC.ERROR
      012110          ENDIF
2920 012110          IF R4 NE R4.EXP THEN          $50404:
      012110 020467 166420          CMP      R4,R4.EXP
      012114 001402          BEQ      $50405
2921 012116          CALL ERROR          JSR      PC.ERROR
      012116 004767 045556          ENDIF
2922 012122          IF R5 NE R5.EXP THEN          $50405:
      012122 020567 166410          CMP      R5,R5.EXP
      012126 001402          BEQ      $50406
2924 012130          CALL ERROR          JSR      PC.ERROR
      012130 004767 045544          ENDIF
2925 012134          LET TSTCAS := TSTCAS + #4          $50406:
      012134 062767 000004 166402          ADD      #4,TSTCAS
2927 012142          LET COUNT := COUNT - #1          DEC      COUNT
      012142 005367 166322          ;UNTIL COUNT EQ #0
2928          IF COUNT NE #0 THEN
2929 012146          INLINE <JMP      $5042>          TST      COUNT
      012146 005767 166316          BEQ      $50407
      012152 001402          JMP      $5042
2930 012154          ENDIF          $50407:
      012154 000167 177326
2931 012160
      012160
2932
2933
2934          ;+
2935          ; DO ONE INLINE CASE - SCANCI.
2936          ;-
2937
2938 012160          LET BFADRS := #BUFF1
      012160 012767 063236 166354          MOV      #BUFF1,BFADRS
2939 012166          CALL CLRBUF          JSR      PC.CLRBUF
      012166 004767 044264
2940 012172          LET TEMPO := #BUFF1 + #50          MOV      #BUFF1,TEMPO
      012172 012767 063236 166276          ADD      #50,TEMPO
      012200 062767 000050 166270
2941 012206          LET @TEMPO :B= #27          MOVB     #27,@TEMPO
      012206 112777 000027 166262
2942 012214          LET R0 := #CHRSET          MOV      #CHRSET,R0
      012214 012700 064036
2943 012220          WHILE R0 LO #CHRSET+400 DO          $50410:
      012220 020027 064436          CMP      R0,#CHRSET-400
      012224 103002          BHS     $50411
2944 012226          INLINE <CLR      (R0)>>          ;CLEAR CHAR.SET TABLE
      012226 005020          CLR      (R0)-
2945 012230          ENDDO          BR      $50410
      012230 000773          $50411:
      012232
2946 012232          LET TEMPO := #CHRSET + #27          MOV      #CHRSET,TEMPO
      012232 012767 064036 166236

```



SCANC

```

2973 012372          IF R5 NE #0 THEN          $50417:
012372 005705
012374 001402
2974 012376          CALL ERROR                TST      R5
012376 004767 045276                                BEQ     $50420
2975 012402          ENDIF                      JSR     PC.ERROR
012402
2976 012402          INLINE <JMP XSPANC>        $50420:
012402 000167 000026                                JMP     XSPANC
2977
2978
2979 ;*
2980 ; TABLE OF TEST CASES FOR SCANC.
2981 ; -
2982 012406 000003  TSCANC: .WORD 3 ;#OF TEST CASES FOR SCANC
2983
2984 012410 000000          .WORD 0 ;SRC.LEN
2985 012412 000020          .WORD 20 ;OFFSET
2986
2987 012414 000100          .WORD BUFLN ;TEST CASE NUMBER 2
2988 012416 000030          .WORD 30
2989
2990 012420 000100          .WORD BUFLN ;TEST CASE NUMBER 3
2991 012422 000110          .WORD BUFLN+10
2992
2993
2994 012424 000100  TSCNCI: .WORD BUFLN ;TEST CASE FOR INLINE
2995 012426 063236          .WORD BUFF1
2996 012430 000010          .WORD 10
2997 012432 064036          .WORD CHRSET
2998

```

SPANC

```

3000 .SBTTL SPANC
3001 ;*****
3002 ;
3003 ; SPANC TEST
3004 ;
3005 ;*****
3006 ;
3007 012434 ROUTINE XSPANC
      012434                                     XSPANC:
3008 012434      INLINE <SCOPE>                                     SCOPE
      012434 000004
3009 012436      INLINE <NOP>                                       NOP
      012436 000240
3010 012440      LET $TIMES := #20 ;ITERATION COUNT
      012440 012767 000020 165736
3011 012446      LET COUNT := TSPANC ;GET TEST COUNT
      012446 016767 000660 166014
3012 012454      LET TSTCAS := #TSPANC+2 ;POINT TO FIRST TEST CASE
      012454 012767 013334 166062
3013 ;REPEAT
3014 012462      INLINE <#5043:>
      012462
3015 ;
3016 ;*
3017 ; GET TEST DATA FROM TABLE
3018 ;-
3019 ;
3020 012462      LET R0 := TSTCAS
      012462 016700 166056
3021 012466      LET DATA00 := (R0)+ ;SRC.LEN
      012466 012067 166070
3022 012472      LET DATA01 := (R0) ;OFFSET
      012472 011067 166066
3023 ;
3024 ;*
3025 ; SET UP GPR'S,
3026 ; EXPECTED GPR'S AND CONDITION CODES,
3027 ; SOURCE BUFFER, AND CHARACTER SET TABLE.
3028 ;-
3029 ;
3030 012476      LET R0 := DATA00
      012476 016700 166060
3031 012502      LET R1 := #BUFF1
      012502 012701 063236
3032 012506      LET R4 := #40 ;MASK
      012506 012704 000040
3033 012512      LET R5 := #CHRSET
      012512 012705 064036
3034 012516      INLINE <BIT #BIT00,COUNT>
      012516 032767 000001 165744
3035 012524      IFCOND EQ THEN
      012524 001003
3036 012526      LET R2 := #0
      012526 005002
3037 012530      LET R3 := #0
      012530 005003
3038 012532      ELSE

```



SPANC

012532	000404								
3039	012534				LET R2 := #177777		\$50423:	BR	\$50424
3040	012534	012702	177777					MOV	#177777,R2
	012540				LET R3 := #177777			MOV	#177777,R3
3041	012540	012703	177777		ENDIF		\$50424:		
3042	012544				IF DATA01 HI DATA00 THEN			CMP	DATA01,DATA00
	012544	026767	166014	166010				BLOS	\$50425
3043	012552	101411			LET R0.EXP := #0			CLR	R0.EXP
3044	012554	005067	165744		LET R1.EXP := #BUFF1 + DATA00			MOV	#BUFF1,R1.EXP
	012560	012767	063236	165740				ADD	DATA00,R1.EXP
3045	012566	066767	165770	165732	ELSE			BR	\$50426
	012574	000414					\$50425:		
3046	012576				LET R0.EXP := DATA00 - DATA01			MOV	DATA00,R0.EXP
	012576	016767	165760	165720				SUB	DATA01,R0.EXP
3047	012604	166767	165754	165712	LET R1.EXP := #BUFF1 + DATA01			MOV	#BUFF1,R1.EXP
	012612	012767	063236	165706				ADD	DATA01,R1.EXP
3048	012620	066767	165740	165700	ENDIF		\$50426:		
	012626				LET R2.EXP := R2			MOV	R2,R2.EXP
3049	012626	010267	165676		LET R3.EXP := R3			MOV	R3,R3.EXP
3050	012632	010367	165674		LET R4.EXP := R4			MOV	R4,R4.EXP
3051	012636	010467	165672		LET R5.EXP := R5			MOV	R5,R5.EXP
3052	012642	010567	165670		INLINE <TST R0.EXP>			TST	R0.EXP
3053	012646	005767	165652		LET CC.EXP := PSW			MOV	PSW,CC.EXP
3054	012652	016767	165120	165642	LET CC.EXP := CC.EXP CLR.BY #177760			BIC	#177760,CC.EXP
3055	012660	042767	177760	165634	LET BFADRS := #BUFF1			MOV	#BUFF1,BFADRS
3056	012666	012767	063236	165646	CALL CLRBUF			JSR	PC,CLRBUF
3057	012674	004767	043556		LET TEMPO := #BUFF1 + DATA01			MOV	#BUFF1,TEMPO
3058	012700	012767	063236	165570				ADD	DATA01,TEMPO
	012700	066767	165652	165562	LET @TEMPO := #252			MOVB	#252,@TEMPO
3059	012714	112777	000252	165554	WHILE R5 LO #CHRSET+400 DO		\$50427:		
3060	012722							CMP	R5,#CHRSET+400
	012722	020527	064436					BHIS	\$50430
	012726	103002			INLINE <CLR (R5)>			CLR	(R5).
3061	012730								
	012730	005025							
3062	012732				ENDDO				



SPANC

3093	013050			IF R4 NE R4.EXP THEN		
	013050	020467	165460			
	013054	001402				
3094	013056			CALL ERROR		
	013056	004767	044616			
3095	013062			ENDIF		
	013062					
3096	013062			IF R5 NE R5.EXP THEN		
	013062	020567	165450			
	013066	001402				
3097	013070			CALL ERROR		
	013070	004767	044604			
3098	013074			ENDIF		
	013074					
3099	013074			LET TSTCAS := TSTCAS + #4		
	013074	062767	000004	165442		
3100	013102			LET COUNT := COUNT - #1		
	013102	005367	165362			
3101				:UNTIL COUNT EQ #0		
3102	013106			IF COUNT NE #0 THEN		
	013106	005767	165356			
	013112	001402				
3103	013114			INLINE <JMP \$5043>		
	013114	000167	177342			
3104	013120			ENDIF		
	013120					
3105						
3106						
3107						
3108				: DO ONE INLINE CASE - SPANCI		
3109				: -		
3110						
3111	013120			LET BFADRS := #BUFF1		
	013120	012767	063236	165414		
3112	013126			CALL CLRBUF		
	013126	004767	043324			
3113	013132			LET TEMPO := #BUFF1 + #50		
	013132	012767	063236	165336		
	013140	062767	000050	165330		
3114	013146			LET @TEMPO :B= #27		
	013146	112777	000027	165322		
3115	013154			LET RO := #CHRSET		
	013154	012700	064036			
3116	013160			WHILE RO LO #CHRSET+400 DO		
	013160					
	013160	020027	064436			
	013164	103002				
3117	013166			LET (RO)+ := #0 ;CLEAR CHAR. SET TABLE		
	013166	005020				
3118	013170			ENDDO		
	013170	000773				
	013172					
3119	013172			CALL CLRREG		
	013172	004767	043242			
3120	013176			LET CHRSET :B= #10		
	013176	112767	000010	050632		
3121	013204			INLINE <SPANCI>		

SPANC				SPANCI			
3122	013204	076143		INLINE	<.WORD TSPNCI>	.WORD	TSPNCI
	013206	013350					
3123	013210			INLINE	<.WORD TSPNCI+4>	.WORD	TSPNCI+4
	013210	013354					
3124	013212			LET	TEMPO := PSW	MOV	PSW,TEMPO
	013212	016767	164560 165256				
3125	013220			INLINE	<BIT #17,TEMPO>	BIT	#17,TEMPO
	013220	032767	000017 165250				
3126	013226			IFCOND	NE THEN	BEQ	\$50443
	013226	001402					
3127	013230			CALL	ERROR	JSR	PC,ERROR
	013230	004767	044444				
3128	013234			ENDIF			
	013234					\$50443:	
3129	013234			LET	TEMPO := #BUFLEN-50	MOV	#BUFLEN 50,TEMPO
	013234	012767	000030 165234				
3130	013242			IF	R0 NE TEMPO THEN	CMP	R0,TEMPO
	013242	020067	165230			BEQ	\$50444
	013246	001402					
3131	013250			CALL	ERROR	JSR	PC,ERROR
	013250	004767	044424				
3132	013254			ENDIF			
	013254					\$50444:	
3133	013254			IF	R1 NE #BUFF1+50 THEN	CMP	R1,#BUFF1+50
	013254	020127	063306			BEQ	\$50445
	013260	001402					
3134	013262			CALL	ERROR	JSR	PC,ERROR
	013262	004767	044412				
3135	013266			ENDIF			
	013266					\$50445:	
3136	013266			IF	R2 NE #0 THEN	TST	R2
	013266	005702				BEQ	\$50446
	013270	001402					
3137	013272			CALL	ERROR	JSR	PC,ERROR
	013272	004767	044402				
3138	013276			ENDIF			
	013276					\$50446:	
3139	013276			IF	R3 NE #0 THEN	TST	R3
	013276	005703				BEQ	\$50447
	013300	001402					
3140	013302			CALL	ERROR	JSR	PC,ERROR
	013302	004767	044372				
3141	013306			ENDIF			
	013306					\$50447:	
3142	013306			IF	R4 NE #0 THEN	TST	R4
	013306	005704				BEQ	\$50450
	013310	001402					
3143	013312			CALL	ERROR	JSR	PC,ERROR
	013312	004767	044362				
3144	013316			ENDIF			
	013316					\$50450:	
3145	013316			IF	R5 NE #0 THEN	TST	R5
	013316	005705				BEQ	\$50451
	013320	001402					
3146	013322			CALL	ERROR	JSR	PC,ERROR
	013322	004767	044352				

SPANC

```

3147 013326          ENDIF
      013326
3148 013326          INLINE <JMP          XCMPC>          $50451:
      013326 000167 000026                                JMP          XCMPC
3149
3150
3151          ;*
3152          ; TABLE OF TEST CASES FOR SPANC
3153          ; -
3154 013332 000003    TSPANC: .WORD 3                ;# OF TEST CASES FOR SPANC
3155
3156 013334 000000    .WORD 0                ;SRC.LEN
3157 013336 000030    .WORD 30               ;OFFSET
3158
3159 013340 000100    .WORD BUFLN          ;TEST CASE NUMBER 2
3160 013342 000040    .WORD 40
3161
3162 013344 000100    .WORD BUFLN          ;TEST CASE NUMBER 3
3163 013346 000104    .WORD BUFLN+4
3164
3165
3166 013350 000100    TSPNCI: .WORD BUFLN          ;TEST CASE FOR INLINE
3167 013352 063236    .WORD BUFF1
3168 013354 000010    .WORD 10
3169 013356 064036    .WORD CHRSET
3170

```

CMPC

```

3172 .SBTTL CMPC
3173 ;*****
3174 ;
3175 ; CMPC TEST
3176 ;
3177 ;*****
3178
3179 013360 ROUTINE XCMPC
013360 XCMPC:
3180 013360 000004 INLINE <SCOPE> SCOPE
013362 000240 INLINE <NOP> NOP
3182 013364 012767 000020 165012 LET $TIMES := #20 ;ITERATION COUNT
013364 012767 000020 165012 LET COUNT := TCMPIC ;GET TEST COUNT
3183 013372 016767 000736 165070 LET TSTCAS := #TCMPIC*2 ;POINT TO FIRST TEST CASE
013400 012767 014336 165136 ;REPEAT
013406 013406 ;REPEAT
013406 013406 ;REPEAT
3187 ;*
3188 ; GET TEST DATA FROM TABLE
3189 ;
3190 ;
3191 ;
3192 013406 LET R0 := TSTCAS
013406 016700 165132 MOV TSTCAS,R0
3193 013412 LET DATA00 := (R0) ;SRC1.LEN
013412 012067 165144 MOV (R0),DATA00
3194 013416 LET DATA01 := (R0) ;SRC2.LEN
013416 012067 165142 MOV (R0),DATA01
3195 013422 LET DATA02 := (R0) ;OFFSET
013422 011067 165140 MOV (R0),DATA02
3196 ;*
3197 ; SET UP GPR'S
3198 ;
3199 ;
3200 ;
3201 013426 LET R0 := DATA00 ;SRC1
013426 016700 165130 MOV DATA00,R0
3202 013432 LET R1 := #BUFF1
013432 012701 063236 MOV #BUFF1,R1
3203 013436 LET R2 := DATA01 ;SRC2
013436 016702 165122 MOV DATA01,R2
3204 013442 LET R3 := #BUFF2
013442 012703 063336 MOV #BUFF2,R3
3205 013446 LET R4 := #377 ;FILL CHARACTER
013446 012704 000377 MOV #377,R4
3206 013452 LET OFFSET := DATA02 ;OFFSET
013452 016767 165110 165032 MOV DATA02,OFFSET
3207 013460 INLINE <BIT #BIT00,COUNT>
013460 032767 000001 165002 BIT #BIT00,COUNT
3208 013466 IFCOND EQ THEN
013466 001002 BNE $50454
3209 013470 LET R5 := #0

```

CMPC

3210	013470	005005			ELSE	CLR	R5
	013472						
	013472	000402				\$50454:	BR \$50455
	013474						
3211	013474				LET R5 := #177777		
	013474	012705	177777				MOV #177777,R5
3212	013500				ENDIF	\$50455:	
	013500						
3213							
3214					::		
3215					:: SET UP BUFFERS		
3216					::-		
3217							
3218	013500				LET BFADRS := #BUFF1		
	013500	012767	063236	165034		MOV	#BUFF1,BFADRS
3219	013506				CALL SETBUF		
	013506	004767	043006			JSR	PC,SETBUF
3220	013512				LET BFADRS := #BUFF2		
	013512	012767	063336	165022		MOV	#BUFF2,BFADRS
3221	013520				CALL SETBUF		
	013520	004767	042774			JSR	PC,SETBUF
3222	013524				IF DATA00 HI DATA01 THEN		
	013524	026767	165032	165032		CMP	DATA00,DATA01
	013532	101412				BLOS	\$50456
3223	013534				LET TEMPO := OFFSET + #BUFF1		
	013534	016767	164752	164734		MOV	OFFSET,TEMPO
	013542	062767	063236	164726		ADD	#BUFF1,TEMPO
3224	013550				LET CC.EXP := #1		
	013550	012767	000001	164744		MOV	#1,CC.EXP
3225	013556				ELSE		
	013556	000411				BR	\$50457
	013560					\$50456:	
3226	013560				LET TEMPO := OFFSET + #BUFF2		
	013560	016767	164726	164710		MOV	OFFSET,TEMPO
	013566	062767	063336	164702		ADD	#BUFF2,TEMPO
3227	013574				LET CC.EXP := #10		
	013574	012767	000010	164720		MOV	#10,CC.EXP
3228	013602				ENDIF		
	013602					\$50457:	
3229	013602				LET @TEMPO := #0		
	013602	105077	164670			CLRB	@TEMPO
3230							
3231					::		
3232					:: SET UP EXPECTED GPR'S AND CONDITION CODES		
3233					::-		
3234							
3235	013606				LET R0.EXP := R0 DATA02		
	013606	010067	164712			MOV	R0,R0.EXP
	013612	166767	164750	164704		SUB	DATA02,R0.EXP
3236	013620				LET R1.EXP := R1 DATA02		
	013620	010167	164702			MOV	R1,R1.EXP
	013624	066767	164736	164674		ADD	DATA02,R1.EXP
3237	013632				LET R2.EXP := R2 DATA02		
	013632	010267	164672			MOV	R2,R2.EXP
	013636	166767	164724	164664		SUB	DATA02,R2.EXP
3238	013644				LET R3.EXP := R3 DATA02		
	013644	010367	164662			MOV	R3,R3.EXP

CMPC

3239	013650	066767	164712	164654	LET R4.EXP := R4	ADD	DATA02,R3.EXP
	013656					MOV	R4,R4.EXP
3240	013662	010467	164652		LET R5.EXP := R5	MOV	R5,R5.EXP
	013662	010567	164650			MOV	R5,R5.EXP
3241	013666				IF DATA02 HIS DATA00 THEN	CMP	DATA02,DATA00
	013666	026767	164674	164666		BLO	\$50460
	013674	103406				CLR	R0.EXP
3242	013676				LET R0.EXP := #0	MOV	R1,R1.EXP
	013676	005067	164622			ADD	R0,R1.EXP
3243	013702				LET R1.EXP := R1 + R0		
	013702	010167	164620				
	013706	060067	164614				
3244	013712				ENDIF		
	013712						
3245	013712				IF DATA02 HIS DATA01 THEN	\$50460:	
	013712	026767	164650	164644		CMP	DATA02,DATA01
	013720	103406				BLO	\$50461
3246	013722				LET R2.EXP := #0	CLR	R2.EXP
	013722	005067	164602			MOV	R3,R3.EXP
3247	013726				LET R3.EXP := R3 + R2	ADD	R2,R3.EXP
	013726	010367	164600				
	013732	060267	164574				
3248	013736				ENDIF		
	013736						
3249	013736				IF DATA02 HIS DATA00 AND DATA02 HIS DATA01 THEN	\$50461:	
	013736	026767	164624	164616		CMP	DATA02,DATA00
	013744	103407				BLO	\$50462
	013746	026767	164614	164610		CMP	DATA02,DATA01
	013754	103403				BLO	\$50462
3250	013756				LET CC.EXP := #4	MOV	#4,CC.EXP
	013756	012767	000004	164536			
3251	013764				ENDIF		
	013764						
3252						\$50462:	
3253					;		
3254					; DO THE CIS INSTRUCTION		
3255					;		
3256							
3257	013764				INLINE <CMPC>		
	013764	076044				CMPC	
3258							
3259					;		
3260					; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS		
3261					;-		
3262							
3263	013766				LET CC := PSW		
	013766	016767	164004	164524		MOV	PSW,CC
3264	013774				LET CC := CC CLR.BY #177760	BIC	#177760,CC
	013774	042767	177760	164516			
3265	014002				IF CC NE CC.EXP THEN	CMP	CC,CC.EXP
	014002	026767	164512	164512		BEQ	\$50463
	014010	001402					
3266	014012				CALL ERROR	JSR	PC.ERROR
	014012	004767	043662				
3267	014016				ENDIF		
	014016						
						\$50463:	



CMPC

```

3268 014016          IF R0 NE R0.EXP THEN
      014016 020067 164502
      014022 001402
3269 014024          CALL ERROR
      014024 004767 043650
3270 014030          ENDIF
3271 014030          IF R1 NE R1.EXP THEN
      014030 020167 164472
      014034 001402
3272 014036          CALL ERROR
      014036 004767 043636
3273 014042          ENDIF
3274 014042          IF R2 NE R2.EXP THEN
      014042 020267 164462
      014046 001402
3275 014050          CALL ERROR
      014050 004767 043624
3276 014054          ENDIF
3277 014054          IF R3 NE R3.EXP THEN
      014054 020367 164452
      014060 001402
3278 014062          CALL ERROR
      014062 004767 043612
3279 014066          ENDIF
3280 014066          IF R4 NE R4.EXP THEN
      014066 020467 164442
      014072 001402
3281 014074          CALL ERROR
      014074 004767 043600
3282 014100          ENDIF
3283 014100          IF R5 NE R5.EXP THEN
      014100 020567 164432
      014104 001402
3284 014106          CALL ERROR
      014106 004767 043566
3285 014112          ENDIF
3286 014112          LET TSTCAS := TSTCAS + #6
      014112 062767 000006 164424
3287 014120          LET COUNT := COUNT - #1
      014120 005367 164314
3288          ;UNTIL COUNT EQ #0
3289 014124          IF COUNT NE #0 THEN
      014124 005767 164340
      014130 001402
3290 014132          INLINE <JMP $5044>
      014132 000167 177250
3291 014136          ENDIF
3292
3293
3294

```

\$50464:

\$50465:

\$50466:

\$50467:

\$50470:

\$50471:

\$50472:

::

CMPC

```

3295 ; DO ONE INLINE CASE CMPCI
3296 :-
3297
3298 014136 LET BFADRS := #BUFF1
014136 012767 063236 164376 MOV #BUFF1,BFADRS
3299 014144 CALL CLRBUF
014144 004767 042306 JSR PC,CLRBUF
3300 014150 LET BFADRS := #BUFF2
014150 012767 063336 164364 MOV #BUFF2,BFADRS
3301 014156 CALL CLRBUF
014156 004767 042274 JSR PC,CLRBUF
3302 014162 CALL CLRREG
014162 004767 042252 JSR PC,CLRREG
3303 014166 LET TEMPO := #BUFF1 * #40
014166 012767 063236 164302 MOV #BUFF1,TEMPO
014174 062767 000040 164274 ADD #40,TEMPO
3304 014202 LET @TEMPO :B= #377
014202 112777 000377 164266 MOVB #377,@TEMPO
3305 014210 INLINE <CMPCI>
014210 076144 CMPCI
3306 014212 INLINE <.WORD TCMPCI>
014212 014416 .WORD TCMPCI
3307 014214 INLINE <.WORD TCMPCI*4>
014214 014422 .WORD TCMPCI*4
3308 014216 INLINE <.WORD 0>
014216 000000 .WORD 0
3309 014220 LET TEMPO := PSW
014220 016767 163552 164250 MOV PSW,TEMPO
3310 014226 LET TEMPO := TEMPO CLR.BY #177760
014226 042767 177760 164242 BIC #177760,TEMPO
3311 014234 IF TEMPO NE #10 THEN
014234 026727 164236 000010 CMP TEMPO,#10
014242 001402 BEQ $50473
3312 014244 CALL ERROR
014244 004767 043430 JSR PC,ERROR
3313 014250 ENDF
014250 $50473:
3314 014250 IF R0 NE #0 THEN
014250 005700 TST R0
014252 001402 BEQ $50474
3315 014254 CALL ERROR
014254 004767 043420 JSR PC,ERROR
3316 014260 ENDF
014260 $50474:
3317 014260 IF R1 NE #0 THEN
014260 005701 TST R1
014262 001402 BEQ $50475
3318 014264 CALL ERROR
014264 004767 043410 JSR PC,ERROR
3319 014270 ENDF
014270 $50475:
3320 014270 IF R2 NE #0 THEN
014270 005702 TST R2
014272 001402 BEQ $50476
3321 014274 CALL ERROR
014274 004767 043400 JSR PC,ERROR
3322 014300 ENDF

```

CMPC

```

3323 014300          IF R3 NE #0 THEN          $50476:
      014300 005703          TST          R3
      014302 001402          BEQ          $50477
3324 014304          CALL ERROR
      014304 004767 043370          JSR          PC,ERROR
3325 014310          ENDIF
3326 014310          IF R4 NE #0 THEN          $50477:
      014310 005704          TST          R4
      014312 001402          BEQ          $50500
3327 014314          CALL ERROR
      014314 004767 043360          JSR          PC,ERROR
3328 014320          ENDIF
3329 014320          IF R5 NE #0 THEN          $50500:
      014320 005705          TST          R5
      014322 001402          BEQ          $50501
3330 014324          CALL ERROR
      014324 004767 043350          JSR          PC,ERROR
3331 014330          ENDIF
3332 014330          INLINE <JMP XMATC>          $50501:
      014330 000167 000072          JMP          XMATC

```

```

3333
3334 ;+
3335 ; TABLE OF TEST CASES FOR CMPC
3336 ;-

```

3337					
3338	014334	000010	TCMPC: .WORD 8.		; # OF TEST CASES FOR CMPC
3339					
3340	014336	000000	.WORD 0		;SRC1.LEN
3341	014340	000020	.WORD 20		;SRC2.LEN
3342	014342	000040	.WORD 40		;OFFSET
3343					
3344	014344	000010	.WORD 10		;TEST CASE NUMBER 2
3345	014346	000030	.WORD 30		
3346	014350	000007	.WORD 7		
3347					
3348	014352	000044	.WORD 44		;TEST CASE NUMBER 3
3349	014354	000070	.WORD 70		
3350	014356	000060	.WORD 60		
3351					
3352	014360	000030	.WORD 30		;TEST CASE NUMBER 4
3353	014362	000050	.WORD 50		
3354	014364	000060	.WORD 60		
3355					
3356	014366	000000	.WORD 0		;TEST CASE NUMBER 5
3357	014370	000000	.WORD 0		
3358	014372	000010	.WORD 10		
3359					
3360	014374	000060	.WORD 60		;TEST CASE NUMBER 6
3361	014376	000040	.WORD 40		
3362	014400	000030	.WORD 30		
3363					
3364	014402	000057	.WORD 57		;TEST CASE NUMBER 7
3365	014404	000033	.WORD 33		

CMPC

3366	014406	000045	.WORD	45	
3367					
3368	014410	000020	.WORD	20	;TEST CASE NUMBER 8
3369	014412	000011	.WORD	11	
3370	014414	000023	.WORD	23	
3371					
3372					
3373	014416	000060	TCMPCI: .WORD	60	;TEST CASE FOR INLINE
3374	014420	063236	.WORD	BUFF1	
3375	014422	000060	.WORD	60	
3376	014424	063336	.WORD	BUFF2	
3377					

MATC

```

3379          .SBTTL  MATC
3380          ;*****
3381          ;
3382          ;   MATC TEST
3383          ;
3384          ;*****
3385
3386 014426    ROUTINE XMATC
3387 014426    000004    XMATC:
3388 014430    000240    ;SCOPE
3389 014432    012767 000020 163744    ;ITERATION COUNT
3390 014440    016767 000670 164022    ;GET TEST COUNT
3391 014446    012767 015336 164070    ;POINT TO FIRST TEST CASE
3392          ;REPEAT
3393 014454    014454    ;   INLINE  <$5045:>
3394          ;
3395          ;*
3396          ; GET TEST DATA FROM TABLE
3397          ;-
3398
3399 014454    016700 164064    LET R0 := TSTCAS
3400 014460    012067 164076    ;OBJ.LEN
3401 014464    011067 164074    ;OFFSET
3402
3403          ;*
3404          ; SET UP THE GPR'S
3405          ;-
3406
3407 014470    012700 000070    LET R0 := #70
3408 014474    012701 063236    ;SRC.LEN
3409 014500    016702 164056    ;SRC.ADR
3410 014504    012703 063336    LET R1 := #BUFF1
3411 014510    032767 000001 163752    LET R2 := DATA00
3412 014516    001003          LET R3 := #BUFF2
3413 014520    005004          ;OBJ.LEN
3414 014522    005005          ;SRC.LEN
3415 014524    000404          ;SRC.ADR
3415 014526          ELSE
3415 014526          ;   INLINE  <BIT #BIT00,COUNT>
3415 014526          IFCOND EQ THEN
3415 014526          ;   BIT    #BIT00,COUNT
3415 014526          ;   BNE   $50504
3415 014526          ;   CLR   R4
3415 014526          ;   CLR   R5
3415 014526          ;   BR    $50505
3415 014526          ;$50504:

```

MATC

```

3416 014526          LET R4 := #177777
      014526 012704 177777
3417 014532          LET R5 := #177777
      014532 012705 177777
3418 014536          ENDIF
      014536
3419
3420
3421          ;*
3422          ; SET UP THE BUFFERS
3423          ; -
3424 014536          LET BFADRS := #BUFF1
      014536 012767 063236 163776
3425 014544          CALL SETBUF
      014544 004767 041750
3426 014550          LET BFADRS := #BUFF2
      014550 012767 063336 163764
3427 014556          CALL CLRBUF
      014556 004767 041674
3428 014562          IF DATA01 LO #70 THEN
      014562 026727 163776 000070
      014570 103015
3429 014572          LET R1 := R1 + DATA01
      014572 066701 163766
3430 014576          REPEAT
      014576
3431 014576          INLINE <CLRB          (R1)+>
      014576 105021
3432 014600          LET TEMPO := #BUFF1 + R0
      014600 012767 063236 163670
      014606 060067 163664
3433 014612          UNTIL R1 EQ TEMPO
      014612 020167 163660
      014616 001367
3434 014620          LET R1 := #BUFF1          ;RESTORE R1
      014620 012701 063236
3435 014624          ENDIF
      014624
3436
3437
3438          ;*
3439          ; SET UP EXPECTED GPR'S AND CONDITION CODES
3440          ; -
3441 014624          LET R0.EXP := R0
      014624 010067 163674
3442 014630          LET R1.EXP := R1
      014630 010167 163672
3443 014634          LET R2.EXP := R2
      014634 010267 163670
3444 014640          LET R3.EXP := R3
      014640 010367 163666
3445 014644          LET R4.EXP := R4
      014644 010467 163664
3446 014650          LET R5.EXP := R5
      014650 010567 163662
3447 014654          IF DATA00 HI #0 THEN
      014654 005767 163702

```

\$50505:

\$50507:

\$50506:

```

MOV      #177777,R4
MOV      #177777,R5
MOV      #BUFF1,BFADRS
JSR      PC,SETBUF
MOV      #BUFF2,BFADRS
JSR      PC,CLRBUF
CMP      DATA01,#70
BHS      $50506
ADD      DATA01,R1
MOV      #BUFF1,TEMPO
ADD      R0,TEMPO
CLRB     (R1)+
MOV      #BUFF1,TEMPO
ADD      R0,TEMPO
CMP      R1,TEMPO
BNE      $50507
MOV      #BUFF1,R1
MOV      R0,R0.EXP
MOV      R1,R1.EXP
MOV      R2,R2.EXP
MOV      R3,R3.EXP
MOV      R4,R4.EXP
MOV      R5,R5.EXP
TST      DATA00

```

MATC

3448	014660	101432				BLOS	\$50510
	014662				LET TEMPO := RO - DATA01	MOV	RO,TEMPO
	014666	010067	163610			SUB	DATA01,TEMPO
3449	014674	166767	163672	163602			
	014674	026767	163576	163660	IF TEMPO LO DATA00 THEN	CMP	TEMPO,DATA00
	014702	103007				BHIS	\$50511
3450	014704				LET RO.EXP := #0	CLR	RO.EXP
	014704	005067	163614				
3451	014710				LET R1.EXP := RO + R1	MOV	RO,R1.EXP
	014710	010067	163612			ADD	R1,R1.EXP
	014714	060167	163606				
3452	014720				ELSE	BR	\$50512
	014720	000412					
	014722					\$50511:	
3453	014722				LET RO.EXP := RO - DATA01	MOV	RO,RO.EXP
	014722	010067	163576			SUB	DATA01,RO.EXP
	014726	166767	163632	163570			
3454	014734				LET R1.EXP := R1 + DATA01	MOV	R1,R1.EXP
	014734	010167	163566			ADD	DATA01,R1.EXP
	014740	066767	163620	163560			
3455	014746				ENDIF		
	014746					\$50512:	
3456	014746				ENDIF		
	014746					\$50510:	
3457	014746				INLINE <TST RO.EXP>		
	014746	005767	163552			TST	RO.EXP
3458	014752				LET CC.EXP := PSW	MOV	PSW,CC.EXP
	014752	016767	163020	163542			
3459	014760				LET CC.EXP := CC.EXP CLR.BY #177760	BIC	#177760,CC.EXP
	014760	042767	177760	163534			
3460							
3461					;		
3462					; DO THE CIS INSTRUCTION		
3463					;		
3464					;-		
3465	014766				INLINE <MATC>		
	014766	076045				MATC	
3466							
3467					;		
3468					; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS		
3469					;		
3470					;-		
3471	014770				LET CC := PSW		
	014770	016767	163002	163522		MOV	PSW,CC
3472	014776				LET CC := CC CLR.BY #177760	BIC	#177760,CC
	014776	042767	177760	163514			
3473	015004				IF CC NE CC.EXP THEN	CMP	CC,CC.EXP
	015004	026767	163510	163510		BEG	\$50513
	015012	001402					
3474	015014				CALL ERROR	JSR	PC,ERROR
	015014	004767	042660				
3475	015020				ENDIF		
	015020					\$50513:	
3476	015020				IF RO NE RO.EXP THEN	CMP	RO,RO.EXP
	015020	020067	163500			BEG	\$50514
	015024	001402					

MATC

```

3477 015026          CALL ERROR
      015026 004767 042646
3478 015032          ENDIF
      015032
3479 015032          IF R1 NE R1.EXP THEN
      015032 020167 163470
      015036 001402
3480 015040          CALL ERROR
      015040 004767 042634
3481 015044          ENDIF
      015044
3482 015044          IF R2 NE R2.EXP THEN
      015044 020267 163460
      015050 001402
3483 015052          CALL ERROR
      015052 004767 042622
3484 015056          ENDIF
      015056
3485 015056          IF R3 NE R3.EXP THEN
      015056 020367 163450
      015062 001402
3486 015064          CALL ERROR
      015064 004767 042610
3487 015070          ENDIF
      015070
3488 015070          IF R4 NE R4.EXP THEN
      015070 020467 163440
      015074 001402
3489 015076          CALL ERROR
      015076 004767 042576
3490 015102          ENDIF
      015102
3491 015102          IF R5 NE R5.EXP THEN
      015102 020567 163430
      015106 001402
3492 015110          CALL ERROR
      015110 004767 042564
3493 015114          ENDIF
      015114
3494 015114          LET TSTCAS := TSTCAS + #4
      015114 062767 000004 163422
3495 015122          LET COUNT := COUNT - #1
      015122 005367 163342
3496          ;UNTIL COUNT EQ #0
3497 015126          IF COUNT NE #0 THEN
      015126 005767 163336
      015132 001402
3498 015134          INLINE <JMP $5045>
      015134 000167 177314
3499 015140          ENDIF
      015140
3500
3501
3502
3503          ; DO ONE INLINE CASE - MATCI
3504
3505

```

\$50514:

```

JSR PC.ERROR
CMP R1,R1.EXP
BEQ $50515

```

\$50515:

```

JSR PC.ERROR
CMP R2,R2.EXP
BEQ $50516

```

\$50516:

```

JSR PC.ERROR
CMP R3,R3.EXP
BEQ $50517

```

\$50517:

```

JSR PC.ERROR
CMP R4,R4.EXP
BEQ $50520

```

\$50520:

```

JSR PC.ERROR
CMP R5,R5.EXP
BEQ $50521

```

\$50521:

```

JSR PC.ERROR
ADD #4,TSTCAS
DEC COUNT

```

\$50522:

```

TST COUNT
BEQ $50522
JMP $5045

```



MATC

3506	015140			LET BFADRS := #BUFF1			
	015140	012767	063236				MOV #BUFF1,BFADRS
3507	015146			CALL SETBUF			
	015146	004767	041346				JSR PC,SETBUF
3508	015152			LET BFADRS := #BUFF2			
	015152	012767	063336				MOV #BUFF2,BFADRS
3509	015160			CALL SETBUF			
	015160	004767	041334				JSR PC,SETBUF
3510	015164			LET R0 := #BUFF1			
	015164	012700	063236				MOV #BUFF1,R0
3511	015170			REPEAT			
	015170						\$50523:
3512	015170			INLINE <CLRB (R0)>			CLRB (R0).
	015170	105020					
3513	015172			UNTIL R0 EQ #BUFF1+40			
	015172	020027	063276				CMP R0,#BUFF1+40
	015176	001374					BNE \$50523
3514	015200			CALL CLRREG			
	015200	004767	041234				JSR PC,CLRREG
3515	015204			INLINE <MATCI>			
	015204	076145					MATCI
3516	015206			INLINE <.WORD T MATCI>			.WORD T MATCI
	015206	015356					
3517	015210			INLINE <.WORD T MATCI+4>			.WORD T MATCI+4
	015210	015362					
3518	015212			LET CC := PSW			
	015212	016767	162560				MOV PSW,CC
3519	015220			INLINE <BIT #17,CC>			
	015220	032767	000017				BIT #17,CC
3520	015226			IFCOND NE THEN			BEQ \$50524
	015226	001402		CALL ERROR			
3521	015230			CALL ERROR			JSR PC,ERROR
	015230	004767	042444	ENDIF			
3522	015234						\$50524:
	015234			IF R0 NE #30 THEN			
	015234	020027	000030	;SRC.LEN - 40			CMP R0,#30
	015240	001402					BEQ \$50525
3524	015242			CALL ERROR			
	015242	004767	042432				JSR PC,ERROR
3525	015246			ENDIF			
	015246						\$50525:
3526	015246			IF R1 NE #BUFF1+40 THEN			
	015246	020127	063276				CMP R1,#BUFF1+40
	015252	001402					BEQ \$50526
3527	015254			CALL ERROR			
	015254	004767	042420				JSR PC,ERROR
3528	015260			ENDIF			
	015260						\$50526:
3529	015260			IF R2 NE #0 THEN			
	015260	005702					TST R2
	015262	001402					BEQ \$50527
3530	015264			CALL ERROR			
	015264	004767	042410				JSR PC,ERROR
3531	015270			ENDIF			
	015270						\$50527:
3532	015270			IF R3 NE #0 THEN			

MATC

```

015270 005703
015272 001402
3533 015274          CALL ERROR
015274 004767 042400
3534 015300          ENDIF
015300
3535 015300          IF R4 NE #0 THEN
015300 005704
015302 001402
3536 015304          CALL ERROR
015304 004767 042370
3537 015310          ENDIF
015310
3538 015310          IF R5 NE #0 THEN
015310 005705
015312 001402
3539 015314          CALL ERROR
015314 004767 042360
3540 015320          ENDIF
015320
3541 015320          IF R2 NE #0 THEN
015320 005702
015322 001402
3542 015324          CALL ERROR
015324 004767 042350
3543 015330          ENDIF
015330
3544 015330          INLINE <JMP INTST>
015330 000167 000032

```

```

TST R3
BEQ #50530
JSR PC,ERROR
$50530:
TST R4
BEQ #50531
JSR PC,ERROR
$50531:
TST R5
BEQ #50532
JSR PC,ERROR
$50532:
TST R2
BEQ #50533
JSR PC,ERROR
$50533:
JMP INTST

```

```

3545
3546 ;*
3547 ; TABLE OF TEST DATA FOR MATC
3548 ;-
3549
3550 TMATC: .WORD 4
3551
3552 .WORD 0
3553 .WORD 10
3554
3555 .WORD 74
3556 .WORD 10
3557
3558 .WORD 40
3559 .WORD 20
3560
3561 .WORD 40
3562 .WORD 50
3563
3564
3565 TMATCI: .WORD 70
3566 .WORD BUFF1
3567 .WORD 10
3568 .WORD BUFF2
3569

```

```

;# OF TEST CASES FOR MATC
;OBJ.LEN
;OFFSET
;TEST CASE NUMBER 2
;TEST CASE NUMBER 3
;TEST CASE NUMBER 4
;TEST CASE FOR INLINE

```

MATC

```

3571      ;:*****
3572      .SBTTL  INTERRUPTABILITY SUBTEST
3573      ;:*****
3574
3575      ;*
3576      ;   THIS SUBTEST VERIFIES THAT A CIS INSTRUCTION CAN BE
3577      ;   SUSPENDED (USING CONSOLE TERMINAL PRINTER INTERRUPT).
3578      ;
3579      ;-
3580 015366 ROUTINE INTST
      015366
3581 015366      INLINE      <SCOPE>                                INTST:
      015366 000004                                SCOPE
3582 015370      INLINE <NOP>                                NOP
      015370 000240
3583 015372      IFB $ENV EQ #1 THEN      ;IF UNDER APT THEN
      015372 126727 163254 000001                                CMPB  $ENV,#1
      015400 001005                                BNE  $50536
3584 015402      IF $PASS NE #0 THEN      ; IF 1ST PASS COMPLETED THEN
      015402 005767 163232                                TST  $PASS
      015406 001402                                BEQ  $50537
3585 015410      INLINE <JMP SKPTST>      ; SKIP THE TEST
      015410 000167 001014                                JMP  SKPTST
3586 015414      ENDIF      ;ELSE
      015414
3587 015414      ENDIF      ; DO THE TEST      $50537:
      015414                                $50536:
3588
3589 015414      LET $TIMES := #1      ;ITERATION COUNT
      015414 012767 000001 162762                                MOV  #1,$TIMES
3590 015422      LET ERR := #2      ;SET UP FOR ERROR MSG
      015422 012767 000002 163122                                MOV  #2,ERR
3591 015430      LET SAVPSW := PSW
      015430 016767 162342 163156                                MOV  PSW,SAVPSW
3592 015436      LET PSW := #PR7
      015436 012767 000340 162332                                MOV  #PR7,PSW
3593 015444      LET TEMPO := #0      ;SUSPENDED FLAG,1=INST SUSPENDED
      015444 005067 163026                                CLR  TEMPO
3594 015450      LET DELAY := #13000
      015450 012767 013000 163014                                MOV  #13000,DELAY
3595
3596      ;*
3597      ;   SET UP TP VECTOR
3598      ;
3599      ;-
3600 015456      LET TPS := #0      ;MAKE SURE INTERRUPT ENABLE IS OFF
      015456 005067 162102                                CLR  TPS
3601 015462      LET TPVEC := #TPSRVC
      015462 012767 016432 162374                                MOV  #TPSRVC,TPVEC
3602 015470      LET TPVEC*2 := #PR7
      015470 012767 000340 162370                                MOV  #PR7,TPVEC*2
3603
3604      ;*
3605      ;   SET UP FOR MOVTC INST
3606      ;
3607      ;-
3608 015476      CALL SAVREG      ;SAVE R0 R5

```

INTERRUPTABILITY SUBTEST

3609	015476	004767	041234			JSR	PC, SAVREG
	015502			REPEAT			
	015502					\$50540:	
3610	015502			LET TEMP1 := DELAY		MOV	DELAY, TEMP1
	015502	016767	162764	162770			
3611	015510			LET DATA00 := #1700	; SRC.LEN	MOV	#1700, DATA00
	015510	012767	001700	163044			
3612	015516			LET DATA01 := #SPACE	; SRC.ADR	MOV	#SPACE, DATA01
	015516	012767	066152	163040			
3613	015524			LET DATA02 := #1740	; DST.LEN	MOV	#1740, DATA02
	015524	012767	001740	163034			
3614	015532			LET DATA03 := #SPACEY	; DST.ADR	MOV	#SPACEY, DATA03
	015532	012767	072152	163030			
3615	015540			LET DATA04 := #0	; FILL CHAR.	CLR	DATA04
	015540	005067	163026				
3616							
3617	015544			LET R0 := DATA00		MOV	DATA00, R0
	015544	016700	163012				
3618	015550			LET R1 := DATA01		MOV	DATA01, R1
	015550	016701	163010				
3619	015554			LET R2 := DATA02		MOV	DATA02, R2
	015554	016702	163006				
3620	015560			LET R3 := DATA03		MOV	DATA03, R3
	015560	016703	163004				
3621	015564			LET R4 := DATA04		MOV	DATA04, R4
	015564	016704	163002				
3622	015570			LET R5 := #TRANS		MOV	#TRANS, R5
	015570	012705	063436				
3623							
3624	015574			LET R0.EXP := #0		CLR	R0.EXP
	015574	005067	162724				
3625	015600			LET R1.EXP := #0		CLR	R1.EXP
	015600	005067	162722				
3626	015604			LET R2.EXP := #0		CLR	R2.EXP
	015604	005067	162720				
3627	015610			LET R3.EXP := #0		CLR	R3.EXP
	015610	005067	162716				
3628	015614			LET R4.EXP := R4		MOV	R4, R4.EXP
	015614	010467	162714				
3629	015620			LET R5.EXP := R5		MOV	R5, R5.EXP
	015620	010567	162712				
3630	015624			LET BFADRS := DATA01		MOV	DATA01, BFADRS
	015624	016767	162734	162710			
3631							
3632	015632			CALL ALTBUI		JSR	PC, ALTBUI
	015632	004767	040772				
3633	015636			LET BFADRS := DATA03		MOV	DATA03, BFADRS
	015636	016767	162726	162676			
3634	015644			CALL ALTBUI		JSR	PC, ALTBUI
	015644	004767	040760				
3635							
3636							
3637				:: SET UP THE TRANSLATION TABLE			
3638				::			
3639				::			
3640	015650			DECR TEMPO FROM #200 TO #1 BY #1		MOV	#200, TEMPO
	015650	012767	000200	162620			

INTERRUPTABILITY SUBTEST

	015656	000402						
	015660							
	015660	005367	162612					
	015664							
	015664	026727	162606	000001				
	015672	002402						
3641	015674				LET (R5). := #0			
	015674	005025						
3642	015676				ENDDC			
	015676	000770						
	015700							
3643	015700				LET R5 := #TRANS			
	015700	012705	063436					
3644	015704				LET (R5) :B= #63			
	015704	112715	000063					
3645	015710				LET 252(R5) :B= #314			
	015710	112765	000314	000252				
3646	015716				LET 377(R5) :B= #125			
	015716	112765	000125	000377				
3647								
3648	015724				WHILE #BIT07 NOTSETIN TPS DO			
	015724							
	015724	032767	000200	161632				
	015732	001001						
3649	015734				ENDDO			
	015734	000773						
	015736							
3650	015736				LET TPB := #0	:SEND CHARACTER TO THE XMIT BUFFER		
	015736	005067	161624			CLR TPB		
3651	015742				INLINE<WAIT:>	:WAIT FOR DONE TO DOUBLE BUFFER		
	015742					WAIT:		
3652	015742				INLINE<BIT #BIT07,TPS>	; XMIT		
	015742	032767	000200	161614				
3653	015750				INLINE<BEQ WAIT>			
	015750	001774						
3654	015752				LET TPB := #0	:FILL SECOND XMIT BUFFFER		
	015752	005067	161610			CLR TPB		
3655	015756				LET TPS := #BIT6	:SET INTERRUPT ENABLE		
	015756	012767	000100	161600				
3656	015764				LET PSW := #PRO	:ALLOW INTERRUPTS		
	015764	012767	000000	162004				
3657	015772				INLINE <LOOP2:>	:DELAY...		
	015772							
3658	015772				INLINE <DEC @#TEMP1>			
	015772	005337	000500					
3659	015776				INLINE <BNE LOOP2>			
	015776	001375						
3660	016000				INLINE <MOVTC>			
	016000	076032						
3661	016002				UNTIL TEMPO EQ #1 OR DELAY LT #40			
	016002	026727	162470	000001				
	016010	001404						
	016012	026727	162454	000040				
	016020	002230						
	016022							
3662	016022				IF TEMPO NE #1 THEN			
	016022	026727	162450	000001				

INTERRUPTABILITY SUBTEST

3663	016030	001403			CALL ERROR				BEQ	\$50547
	016032								SET	
	016032	004767	041642						JSR	PC.ERROR
3664	016036				ELSE					
	016036	000555							BR	\$50550
	016040									\$50547:
3665					;					
3666					;	CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS				
3667					;					
3668					;-					
3669	016040				LET CC := PSW					
	016040	016767	161732	162452					MOV	PSW.CC
3670	016046				LET CC := CC CLR.BY #177760					
	016046	042767	177760	162444					BIC	#177760.CC
3671	016054				IF CC NE CC.EXP THEN					
	016054	026767	162440	162440					CMP	CC.CC.EXP
	016062	001402							BEQ	\$50551
3672	016064				CALL ERROR					
	016064	004767	041610						JSR	PC.ERROR
3673	016070				ENDIF					
	016070									\$50551:
3674	016070				IF R0 NE R0.EXP THEN					
	016070	020067	162430						CMP	R0.R0.EXP
	016074	001402							BEQ	\$50552
3675	016076				CALL ERROR					
	016076	004767	041576						JSR	PC.ERROR
3676	016102				ENDIF					
	016102									\$50552:
3677	016102				IF R1 NE R1.EXP THEN					
	016102	020167	162420						CMP	R1.R1.EXP
	016106	001402							BEQ	\$50553
3678	016110				CALL ERROR					
	016110	004767	041564						JSR	PC.ERROR
3679	016114				ENDIF					
	016114									\$50553:
3680	016114				IF R2 NE R2.EXP THEN					
	016114	020267	162410						CMP	R2.R2.EXP
	016120	001402							BEQ	\$50554
3681	016122				CALL ERROR					
	016122	004767	041552						JSR	PC.ERROR
3682	016126				ENDIF					
	016126									\$50554:
3683	016126				IF R3 NE R3.EXP THEN					
	016126	020367	162400						CMP	R3.R3.EXP
	016132	001402							BEQ	\$50555
3684	016134				CALL ERROR					
	016134	004767	041540						JSR	PC.ERROR
3685	016140				ENDIF					
	016140									\$50555:
3686	016140				IF R4 NE R4.EXP THEN					
	016140	020467	162370						CMP	R4.R4.EXP
	016144	001402							BEQ	\$50556
3687	016146				CALL ERROR					
	016146	004767	041526						JSR	PC.ERROR
3688	016152				ENDIF					
	016152									\$50556:
3689	016152				IF R5 NE R5.EXP THEN					

INTERRUPTABILITY SUBTEST

016152 020567 162360  
 016156 001402  
 3690 016160  
 016160 004767 041514  
 3691 016164  
 016164  
 3692  
 3693  
 3694  
 3695  
 3696  
 3697 016164  
 016164 016700 162400  
 3698 016170  
 016170 117702 162370  
 3699 016174  
 016174 042702 177400  
 3700 016200  
 016200 116201 063436  
 3701 016204  
 016204 005067 162246  
 3702 016210  
 016210 016703 162346  
 3703 016214  
 016214  
 016214 005703  
 016216 001411  
 016220 005767 162232  
 016224 001006  
 3704 016226  
 016226 122001  
 016230 001402  
 3705 016232  
 016232 004767 041442  
 3706 016236  
 016236  
 3707 016236  
 016236 005303  
 3708 016240  
 016240 000765  
 016242  
 3709 016242  
 016242 016700 162322  
 016246 066700 162310  
 3710 016252  
 016252 016703 162310  
 016256 166703 162300  
 3711 016262  
 016262  
 3712 016262  
 016262 122067 162304  
 016266 001402  
 3713 016270  
 016270 004767 041404  
 3714 016274  
 016274  
 3715 016274

```

CALL ERROR
ENDIF

; *
; CHECK ENTIRE DESTINATION BUFFER FOR CORRECT RESULTS
; -

LET R0 := DATA03
LET R2 := @DATA01
LET R2 := R2 CLR.BY #177400
LET R1 := TRANS(R2)
LET ERRFLG := #
LET R3 := DATA00
WHILE R3 NE #0 AND ERRFLG EQ #0 DO

  IFB (R0) NE R1 THEN
    CALL ERROR
  ENDIF
  LET R3 := R3 - #1
ENDDO

LET R0 := DATA03 + DATA00
LET R3 := DATA02 - DATA00
REPEAT
  IFB (R0) NE DATA04 THEN
    CALL ERROR
  ENDIF
  LET R3 := R3 - #1

```

```

CMP R5,R5,EXP
BEQ $50557

JSR PC,ERROR

$50557:
MOV DATA03,R0
MOVB @DATA01,R2
BIC #177400,R2
MOVB TRANS(R2),R1
CLR ERRFLG
MOV DATA00,R3

$50560:
TST R3
BEQ $50561
TST ERRFLG
BNE $50561

CMPB (R0),R1
BEQ $50562

JSR PC,ERROR

$50562:
DEC R3
BR $50560

$50561:
MOV DATA03,R0
ADD DATA00,R0
MOV DATA02,R3
SUB DATA00,R3

$50563:
CMPB (R0),DATA04
BEQ $50564

JSR PC,ERROR

$50564:

```

## INTERRUPTABILITY SUBTEST

3716	016274	005303				DEC	R3
	016276			UNTIL R3 EQ #0 OR ERRFLG NE #0		TST	R3
	016300	005703				BEQ	\$50565
	016302	001403				TST	ERRFLG
	016306	005767	162150			BEQ	\$50563
	016310	001765					
3717	016310			IF DATA02 LO #BUFLEN THEN	\$50565:		
	016310	026727	162252	000100		CMP	DATA02,#BUFLEN
	016316	103025				BHIS	\$50566
3718	016320			LET TEMPO := DATA03 + #1777		MOV	DATA03,TEMPO
	016320	016767	162244	162150		ADD	#1777,TEMPO
	016326	062767	001777	162142			
3719	016334			LET R0 := DATA03 + DATA02		MOV	DATA03,R0
	016334	016700	162230			ADD	DATA02,R0
	016340	066700	162222				
3720	016344			REPEAT	\$50567:		
	016344						
3721	016344			IFB (R0)+ NE BF.EXP THEN		CMPB	(R0)+,BF.EXP
	016344	122067	162170			BEQ	\$50570
	016350	001402					
3722	016352			CALL ERROR		JSR	PC,ERROR
	016352	004767	041322				
3723	016356			ENDIF	\$50570:		
	016356						
3724	016356			UNTIL R0 EQ TEMPO OR ERRFLG NE #0		CMP	R0,TEMPO
	016356	020067	162114			BEQ	\$50571
	016362	001403				TST	ERRFLG
	016364	005767	162066			BEQ	\$50567
	016370	001765					
	016372				\$50571:		
3725	016372			ENDIF	\$50566:		
	016372				\$50550:		
3726	016372			ENDIF			
	016372						
3727	016372			LET PSW := #PR7		MOV	#PR7,PSW
	016372	012767	000340	161376			
3728	016400			LET TPVEC := #TPVEC+2		MOV	#TPVEC+2,TPVEC
	016400	012767	000066	161456			
3729	016406			LET TPVEC+2 := #0 ;RESTORE CONSOLE TRAP CATCHER		CLR	TPVEC+2
	016406	005067	161454			CLR	TPS
3730	016412			LET TPS := #0 ;MAKE SURE INTERRUPT ENABLE IS OFF			
	016412	005067	161146				
3731	016416			CALL RESREG		JSR	PC,RESREG
	016416	004767	040346				
3732	016422			LET PSW := SAVPSW		MOV	SAVPSW,PSW
	016422	016767	162166	161346			
3733	016430			INLINE <SKPTST:>		SKPTST:	
	016430						
3734	016430			INLINE <BR ILLTST> ;BR AROUND INT SERVICE RTN		BR	ILLTST
	016430	000421					
3735							
3736							
3737				::			
3738				::			
3739				::			
3740	016432			ROUTINE TPSRVC			



INTERRUPTABILITY SUBTEST

```

016432
3741 016432          LET TPS := #0
                                ;TURN OFF INTERRUPT ENABLE
016432 005067 161126          ;TPSRVC:
                                CLR      TPS
3742 016436          LET TEMPS := 2(SP)
016436 016667 000002 162044          ;
                                MOV      2(SP),TEMPS
3743 016444          IF #BIT8 SETIN TEMPS THEN
016444 032767 000400 162036          ;
016452 001404          ;
                                BIT      #BIT8,TEMPS
3744 016454          LET TEMPO := #1          ;INST WAS SUSPENDED
016454 012767 000001 162014          ;
                                BEQ      $50574
3745 016462          ELSE
016462 000403          ;
                                MOV      #1,TEMPO
016464          ;
                                BR       $50575
3746 016464          LET DELAY := DELAY - #30 ;DECREMENT DELAY COUNTER AND TRY AGAIN
016464 162767 000030 162000          ;
                                SUB      #30,DELAY
3747 016472          ENDIF
016472          ;
3748 016472          INLINE <RTI>
016472 000002          ;
                                RTI
3749

```

INTERRUPTABILITY SUBTEST

```

3751 ;:*****
3752 .SBTTL  ILLEGAL OPCODE SUBTEST
3753 ;:*****
3754
3755 ;*
3756 ;   THIS SUBTEST VERIFIES THAT AN ILLEGAL CIS CLASS OPCODE
3757 ;   (ILLOP=76033) CAUSES A RESERVED INSTRUCTION TRAP (TRAP 10)
3758 ;-
3759
3760 ROUTINE ILLTST
3761 016474          INLINE      <SCOPE>                      ILLTST:
3762 016474 000004          SCOPE
3763 016476 000240          INLINE <NOP>                      NOP
3764 016500 012767 000020 161676  LET $TIMES := #20          ;ITERATION COUNT
3765 016506 005067 161744  LET ERRFLG := #0          ;RESET ERROR FLAG
3766 016512 012767 000003 162032  LET ERR := #3           ;ERROR MSG #
3767 016520 016767 161252 162066  LET SAVPSW := PSW      ;LOCK OUT INTERRUPTS
3768 016526 012767 000340 161242  LET PSW := #PR7
3769 016534 012767 016562 161246  LET RESVEC := #RESRTN  ;DO ILLEGAL CODE
3770 016542 076033          INLINK <ILLOP>
3771 016544 004767 041130          CALL ERROR          ;DIDNT TRAP, ERROR
3772 016550 012767 056406 161232  LET RESVEC := #ILLRES  ;RESTORE RESERVED INSTRUCTION TRAP VECTOR
3773 016556 000167 000040          INLINK <JMP CIS4END>  ;MODULE EXIT
3774
3775 ;*
3776 ;   TRAP HANDLER
3777 ;-
3778 ROUTINE RESRTN
3779 016562          INLINE <NOP>                      RESRTN:
3780 016564 000240          IF ERRFLG EQ #0 THEN          NOP
3781 016572 005767 161666          POP TEMPO,TEMPO          ;REMOVE PC,PSW PUSHED BY TRAP
3782 016576 012667 161700          ;REMOVE PC,PSW PUSHED BY TRAP
3783 016602 012667 161674          ;REMOVE PC,PSW PUSHED BY TRAP
3784 016602          ENDIF
3785 016602          ;ENSURE ERROR FLAG IS RESET          $50602:
3786 016606 005067 161650          LET ERRFLG := #0          CLR      ERRFLG
3787 016606 016767 162002 161162  LET PSW := SAVPSW      MOV      SAVPSW,PSW
3788 016614          LET RESVEC := #ILLRES          ;RESTORE RESERVED INSTRUCTION TRAP VECTOR

```

ILLEGAL OP CODE SUBTEST

3786 016614 012767 056406 161166  
016622  
3787 016622  
3788

INLINE <CIS4END:>

MOV #ILLRES,RESVEC

CIS4END:

ILLEGAL OPCODE SUBTEST

```

3790 ;:*****
3791 ;:*****
3792 ;**
3793 ;           C I S S   M O D U L E
3794 ;
3795 ; THIS MODULE TESTS THE ARITHMETIC CONVERT INSTRUCTIONS FOR
3796 ; PACKED, NUMERIC, AND LONG INTEGER DATA TYPES: CVTLP, CVTLN,
3797 ; CVTPL, CVTNL, CVTPN, AND CVTNP.
3798 ;--
3799 ;:*****
3800 ;:*****
3801
3802
3803 .SBTTL  C I S S  I N I T I A L I Z A T I O N
3804 ;:*****
3805 ;
3806 ;   C I S S  I N I T I A L I Z A T I O N
3807 ;
3808 ;:*****
3809
3810 ;IF SWR<5:0> ARE CLEARED OR IF SWR<1> IS SET THEN EXECUTE
3811 ; THE CIS5 MODULE; OTHERWISE, SKIP TO THE NEXT CIS MODULE.
3812
3813 016622      INLINE  <SCOPE>
3814 016622      000004
3815 016624      104406
3816 016626      017767  161574  161642
3817 016626      042767  177700  161634
3818 016634      005767  161630
3819 016642      001406
3820 016646      001406
3821 016650      032767  000002  161620
3822 016656      001002
3823 016660      000167  013736
3824 016664      000167  013736
3825 016664      000167  013736
3826 016664      000167  013736
3827 016664      000167  013736
3828 016664      000167  013736
3829 016664      000167  013736
3830 016664      000167  013736
3831 016664      000167  013736
3832 016664      000167  013736
3833 016664      000167  013736
3834 016664      000167  013736
3835 016664      000167  013736
3836 016664      000167  013736
3837 016664      000167  013736
3838 016664      000167  013736
3839 016664      000167  013736
3840 016664      000167  013736
3841 016664      000167  013736
3842 016664      000167  013736
3843 016664      000167  013736
3844 016664      000167  013736
3845 016664      000167  013736
3846 016664      000167  013736
3847 016664      000167  013736
3848 016664      000167  013736
3849 016664      000167  013736
3850 016664      000167  013736
3851 016664      000167  013736
3852 016664      000167  013736
3853 016664      000167  013736
3854 016664      000167  013736
3855 016664      000167  013736
3856 016664      000167  013736
3857 016664      000167  013736
3858 016664      000167  013736
3859 016664      000167  013736
3860 016664      000167  013736
3861 016664      000167  013736
3862 016664      000167  013736
3863 016664      000167  013736
3864 016664      000167  013736
3865 016664      000167  013736
3866 016664      000167  013736
3867 016664      000167  013736
3868 016664      000167  013736
3869 016664      000167  013736
3870 016664      000167  013736
3871 016664      000167  013736
3872 016664      000167  013736
3873 016664      000167  013736
3874 016664      000167  013736
3875 016664      000167  013736
3876 016664      000167  013736
3877 016664      000167  013736
3878 016664      000167  013736
3879 016664      000167  013736
3880 016664      000167  013736
3881 016664      000167  013736
3882 016664      000167  013736
3883 016664      000167  013736
3884 016664      000167  013736
3885 016664      000167  013736
3886 016664      000167  013736
3887 016664      000167  013736
3888 016664      000167  013736
3889 016664      000167  013736
3890 016664      000167  013736
3891 016664      000167  013736
3892 016664      000167  013736
3893 016664      000167  013736
3894 016664      000167  013736
3895 016664      000167  013736
3896 016664      000167  013736
3897 016664      000167  013736
3898 016664      000167  013736
3899 016664      000167  013736
3900 016664      000167  013736

```

CVTLP

```

3827 .SBTTL CVTLP
3828 ;*****
3829 ;
3830 ; CVTLP TEST
3831 ;
3832 ;*****
3833 ;
3834 016712 ROUTINE XCVTLP
      016712
3835 016712          INLINE      <SCOPE>          XCVTLP:
      016712 000004          SCOPE
3836 016714          INLINE      <NOP>          NOP
      016714 000240          ;ITERATION COUNT
3837 016716          LET $TIMES := #20          ;GET TEST COUNT
      016716 012767 000020 161460          MOV      #20,$TIMES
3838 016724          LET COUNT := TCVTLP          ;POINT TO FIRST TEST CASE
      016724 016767 001274 161536          MOV      TCVTLP,COUNT
3839 016732          LET TSTCAS := #TCVTLP+2          MOV      #TCVTLP-2,TSTCAS
      016732 012767 020226 161604
3840 ;REPEAT
3841 016740          INLINE      <#5077:>          $5077:
      016740
3842
3843 ;*
3844 ; GET TEST DATA FROM TABLE
3845 ;-
3846
3847 016740          LET R0 := TSTCAS
      016740 016700 161600          MOV      TSTCAS,R0
3848 016744          LET DATA00 := (R0)+          ;DST.DSCR          MOV      (R0)+,DATA00
      016744 012067 161612          ;LONG INTEGER HI          MOV      (R0)+,DATA01
3849 016750          LET DATA01 := (R0)+          ;LONG INTEGER LO          MOV      (R0)+,DATA02
      016750 012067 161610          MOV      (R0)+,DATA02
3850 016754          LET DATA02 := (R0)+
      016754 012067 161606
3851
3852 ;*
3853 ; PUT EXPECTED RESULTS INTO DS.EXP AND CC.EXP
3854 ;-
3855
3856 016760          CALL CLRDSB          ;CLEAR DECIMAL STRING BUFFER AREAS
      016760 004767 037710          JSR      PC,CLRDSB
3857 016764          LET R2 := DATA01          MOV      DATA01,R2
      016764 016702 161574          MOV      DATA02,R3
3858 016770          LET R3 := DATA02          ;CONVERT LONG INTEGER TO AN INTERMEDIATE FORM
      016770 016703 161572          JSR      PC,CVTLI
3859 016774          CALL CVTLI
      016774 004767 012304          CLR      CC.EXP
3860 017000          LET CC.EXP := #0
      017000 005067 161516          ;GET DST.LEN          CLR      CC.EXP
3861 017004          LET DSLEN2 := DATA00 CLR.BY #177740          MOV      DATA00,DSLEN2
      017004 016767 161552 161504          BIC      #177740,DSLEN2
      017012 042767 177740 161476
3862 017020          IF DSLEN2 EQ #0 THEN          ;DST.LEN = 0          TST      DSLEN2
      017020 005767 161472          BNE      $50607
      017024 001014
3863 017026          LET CC.EXP := CC.EXP SET.BY #ZBIT

```

CVTLP

```

3864 017026 052767 000004 161466          LET DS.EXP :B= #14          BIS      #ZBIT,CC.EXP
      017034 112767 000014 045544          IF DSLEN1 HI #0 THEN      MOVB     #14,DS.EXP
3865 017042 005767 161446          LET CC.EXP := CC.EXP SET.BY #VBIT      TST      DSLEN1
      017046 101403          ENDIF                          BLOS     $50610
3866 017050 052767 000002 161444          ENDIF                          BIS      #VBIT,CC.EXP
3867 017056          ENDIF                          $50610:
3868 017056          IF DSLEN2 NE #0 THEN          ;DST.LEN > 0
      017056 005767 161434          ;SRC.LEN = 0
      017062 001577          TST      DSLEN2
3870 017064          IF DSLEN1 EQ #0 THEN          BEQ      $50611
      017064 005767 161424          TST      DSLEN1
      017070 001017          BNE      $50612
3871 017072          LET TEMPO := DSLEN2          MOV      DSLEN2,TEMPO
      017072 016767 161420 161376          INLINE <ASR TEMPO>
3872 017100          LET RO := #DS.EXP + TEMPO          ASR     TEMPO
      017100 006267 161372          MOV      #DS.EXP,RO
3873 017104          LET (RO) :B= #14          ADD     TEMPO,RO
      017104 012700 064606          MOVB     #14,(RO)
      017110 066700 161362          LET CC.EXP := CC.EXP SET.BY #ZBIT      BIS      #ZBIT,CC.EXP
3874 017114          ELSE
      017114 112710 000014          BR       $50613
3875 017120          IF DSLEN2 EQ DSLEN1 THEN          ;DST.LEN = SRC.LEN > 0
      017120 052767 000004 161374          CMP      DSLEN2,DSLEN1
      017126 000555          BNE      $50614
      017130          LET R1 := #DSBUF1 + #10. - DSLEN1 ;SRC POINTER
      017130 026767 161362 161356          MOV      #DSBUF1,R1
      017136 001010          ADD     #10.,R1
3878 017140          LET R2 := #DS.EXP          ;DST POINTER
      017140 012701 064440          SUB     DSLEN1,R1
      017144 062701 000012          MOV      #DS.EXP,R2
      017150 166701 161340          ENDIF
3879 017154          IF DSLEN2 LT DSLEN1 THEN          ;DST.LEN < SRC.LEN
      017154 012702 064606          CMP      DSLEN2,DSLEN1
3880 017160          LET R1 := #DSBUF1 + #10. - DSLEN2          BGE      $50615
      017160          MOV      #DSBUF1,R1
3881 017160          LET R2 := #DS.EXP          ADD     #10.,R1
      017160 026767 161332 161326          SUB     DSLEN2,R1
      017166 002016          MOV      #DS.EXP,R2
3882 017170          LET CC.EXP := CC.EXP SET.BY #VBIT      BIS      #VBIT,CC.EXP
      017170 012701 064440          LET DSLEN1 := DSLEN2
      017174 062701 000012          MOV      DSLEN2,DSLEN1
      017200 166701 161312          ENDIF
3883 017204          LET CC.EXP := CC.EXP SET.BY #VBIT
      017204 012702 064606          LET DSLEN1 := DSLEN2
3884 017210          LET DSLEN1 := DSLEN2
      017210 052767 000002 161304          BIS      #VBIT,CC.EXP
3885 017216          LET DSLEN1 := DSLEN2
      017216 016767 161274 161270          MOV      DSLEN2,DSLEN1
3886 017224          ENDIF

```

CVTLP

```

3887 017224          IF DSLEN2 GT DSLEN1 THEN          ;DST.LEN > SRC.LEN
      017224 026767 161266 161262                    ;$50615:
      017232 003426                                CMP      DSLEN2,DSLEN1
3888 017234          LET TEMPO := DSLEN1                BLE      $50616
      017234 016767 161254 161234                    MOV      DSLEN1,TEMPO
3889 017242          INLINE <ASR TEMPO>                ASR      TEMPO
      017242 006267 161230
3890 017246          LET TEMP1 := DSLEN2                MOV      DSLEN2,TEMP1
      017246 016767 161244 161224                    ASR      TEMP1
3891 017254          INLINE <ASR TEMP1>                ASR      TEMP1
      017254 006267 161220
3892 017260          LET R2 := #DS.EXP * TEMP1 - TEMPO  MOV      #DS.EXP,R2
      017260 012702 064606                                ADD      TEMP1,R2
      017264 066702 161210                                SUB      TEMPO,R2
      017270 166702 161202
3893 017274          LET R1 := #DSBUF1 * #10. - DSLEN1 ;SRC POINTER
      017274 012701 064440                                MOV      #DSBUF1,R1
      017300 062701 000012                                ADD      #10.,R1
      017304 166701 161204                                SUB      DSLEN1,R1
3894 017310          ENDIF
      017310
3895
3896          ;LOAD REMAINING DIGITS INTO DS.EXP
3897
3898 017310          IF DSLEN1 HI #1 THEN
      017310 026727 161200 000001
      017316 101431
3899 017320          IF #1 NOTSETIN DSLEN1 THEN
      017320 032767 000001 161166
      017326 001003
3900 017330          LET (R2)* := (R1)*                ;OUTPUT TO DEST. BUFFER
      017330 112122                                MOV      (R1)*,(R2)*
3901 017332          LET DSLEN1 := DSLEN1 - #1
      017332 005367 161156
3902 017336          ENDIF
      017336
3903 017336          LET R0 := #0                        ;$50620:
      017336 005000                                ;USE R0 TO ASSEMBLE DATA
3904 017340          WHILE DSLEN1 HI #1 DO
      017340
      017340 026727 161150 000001                    ;$50621:
      017346 101415                                CMP      DSLEN1,#1
3905 017350          LET R0 := (R1)*                    BLOS    $50622
      017350 112100                                MOV      (R1)*,R0
3906 017352          INLINE <FSH #4,R0>
      017352 072027 000004
3907 017356          LET TEMPO := R0 OR (R1)*          ;OUTPUT TO DST. BUFFER
      017356 110067 161114                                ASH     #4,R0
      017362 152167 161110                                MOV      R0,TEMPO
3908 017366          LET (R2)* := TEMPO                BIS     (R1)*,TEMPO
      017366 116722 161104                                MOV      TEMPO,(R2)*
3909 017372          LET DSLEN1 := DSLEN1 - #2
      017372 162767 000002 161114
3910 017400          ENDDO
      017400 000757
      017402

```

;\$50622:

CVTLP

```

3911 017402          ENDIF
      017402
3912 017402          LET R0 :B= (R1)+          $50617:
      017402 112100
3913 017404          INLINE <ASH #4,R0>          MOVB      (R1),R0
      017404 072027 000004
3914 017410          IF DATA00 HIS #70000 THEN          ASH #4,R0
      017410 026727 161146 070000          ;*** UNSIGNED PACKED ***
      017416 103404          CMP      DATA00,#70000
3915 017420          LET (R2) :B= R0 OR #17          BLO      $50623
      017420 110012
      017422 152712 000017          MOVB      R0,(R2)
3916 017426          ELSE          BISB      #17,(R2)
      017426 000415          BR      $50624
      017430
3917 017430          IF SIGN0 EQ #0 THEN          $50623:
      017430 005767 161014          ;*** SIGNED PACKED ***
      017434 001004          TST      SIGN0
3918 017436          LET (R2) :B= R0 OR #14          ;SIGN IS +          BNE      $50625
      017436 110012
      017440 152712 000014          MOVB      R0,(R2)
3919 017444          ELSE          BISB      #14,(R2)
      017444 000406          BR      $50626
      017446
3920 017446          LET (R2) :B= R0 OR #15          $50625:
      017446 110012          ;SIGN IS -          MOVB      R0,(R2)
      017450 152712 000015          BISB      #15,(R2)
3921 017454          LET CC.EXP := CC.EXP SET.BY #NBIT          BIS      #NBIT,CC.EXP
      017454 052767 000010 161040
3922 017462          ENDIF
3923 017462          ENDIF          $50626:
      017462          ENDIF          $50624:
3924 017462          ENDIF          $50613:
      017462          ENDIF          $50611:
3925 017462
3926
3927
3928          ;+
3929          ; SET UP GPR'S AND EXPECTED GPR'S.
3930          ; -
3931 017462          LET R0 := DATA00
      017462 016700 161074
3932 017466          LET R1 := #DS.DST          ;DST. ADR          MOV      DATA00,R0
      017466 012701 064544          MOV      #DS.DST,R1
3933 017472          LET R2 := DATA01          MOV      DATA01,R2
      017472 016702 161066
3934 017476          LET R3 := DATA02          MOV      DATA02,R3
      017476 016703 161064
3935 017502          IF #1 SETIN COUNT THEN          MOV      DATA02,R3
      017502 032767 000001 160760          BIT      #1,COUNT
      017510 001405          BEQ      $50627
3936 017512          LET R4 := #177777          MOV      #177777,R4
      017512 012704 177777
3937 017516          LET R5 := #177777          MOV      #177777,R5
      017516 012705 177777

```



CVTLP

3938	017522			ELSE			
	017522	000402					
	017524						
3939	017524			LET R4 := #0	\$50627:	BR	\$50630
	017524	005004				CLR	R4
3940	017526			LET R5 := #0		CLR	R5
	017526	005005					
3941	017530			ENDIF	\$50630:		
	017530						
3942	017530			LET R0.EXP := R0		MOV	R0,R0.EXP
	017530	010067	160770			MOV	R1,R1.EXP
3943	017534			LET R1.EXP := R1			
	017534	010167	160766			CLR	R2.EXP
3944	017540			LET R2.EXP := #0		CLR	R3.EXP
	017540	005067	160764			MOV	R4,R4.EXP
3945	017544			LET R3.EXP := #0		MOV	R5,R5.EXP
	017544	005067	160762				
3946	017550			LET R4.EXP := R4			
	017550	010467	160760				
3947	017554			LET R5.EXP := R5			
	017554	010567	160756				
3948							
3949							
3950				;* : DO THE CIS INSTRUCTION			
3951				:			
3952				:			
3953	017560			INLINE <CVTLP>			
	017560	076077				CVTLP	
3954							
3955				;* : CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.			
3956				:			
3957				:-			
3958							
3959	017562			LET CC := PSW		MOV	PSW,CC
	017562	016767	160210			BIC	#177760,CC
3960	017570			LET CC := CC CLR.BY #177760			
	017570	042767	177760			CMP	CC,CC.EXP
3961	017576			IF CC NE CC.EXP THEN		BEQ	\$50631
	017576	026767	160716			JSR	PC.ERROR
	017604	001402	160716				
3962	017606			CALL ERROR			
	017606	004767	040066				
3963	017612			ENDIF	\$50631:		
	017612						
3964	017612			IF R0 NE R0.EXP THEN		CMP	R0,R0.EXP
	017612	020067	160706			BEQ	\$50632
	017616	001402				JSR	PC.ERROR
3965	017620			CALL ERROR			
	017620	004767	040054				
3966	017624			ENDIF	\$50632:		
	017624						
3967	017624			IF R1 NE R1.EXP THEN		CMP	R1,R1.EXP
	017624	020167	160676			BEQ	\$50633
	017630	001402				JSR	PC.ERROR
3968	017632			CALL ERROR			
	017632	004767	040042				
3969	017636			ENDIF			

CVTLP

3970	017636			IF R2 NE R2.EXP THEN	\$50633:		
	017636	020267	160666			CMP	R2,R2.EXP
	017642	001402				BEQ	\$50634
3971	017644			CALL ERROR		JSR	PC.ERROR
	017644	004767	040030				
3972	017650			ENDIF	\$50634:		
	017650						
3973	017650			IF R3 NE R3.EXP THEN		CMP	R3,R3.EXP
	017650	020367	160656			BEQ	\$50635
	017654	001402					
3974	017656			CALL ERROR		JSR	PC.ERROR
	017656	004767	040016				
3975	017662			ENDIF	\$50635:		
	017662						
3976	017662			IF R4 NE R4.EXP THEN		CMP	R4,R4.EXP
	017662	020467	160646			BEQ	\$50636
	017666	001402					
3977	017670			CALL ERROR		JSR	PC.ERROR
	017670	004767	040004				
3978	017674			ENDIF	\$50636:		
	017674						
3979	017674			IF R5 NE R5.EXP THEN		CMP	R5,R5.EXP
	017674	020567	160636			BEQ	\$50637
	017700	001402					
3980	017702			CALL ERROR		JSR	PC.ERROR
	017702	004767	037772				
3981	017706			ENDIF	\$50637:		
	017706						
3982							
3983							
3984				;;			
3985				;; COMPARE THE EXPECTED DECIMAL STRING BUFFER			
3986				;; WITH ACTUAL RESULTS.			
3987				;;			
3988	017706			LET R0 := #DS.DST		MOV	#DS.DST,R0
	017706	012700	064544				
3989	017712			LET R1 := #DS.EXP		MOV	#DS.EXP,R1
	017712	012701	064606				
3990	017716			LET ERRFLG := #0		CLR	ERRFLG
	017716	005067	160534				
3991	017722			WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO	\$50640:		
	017722						
	017722	020027	064604			CMP	R0,#DS.DST+40
	017726	001410				BEQ	\$50641
	017730	005767	160522			TST	ERRFLG
	017734	001005				BNE	\$50641
3992	017736			IFB (R0) NE (R1) THEN		CMPB	(R0),(R1)
	017736	122021				BEQ	\$50642
	017740	001402					
3993	017742			CALL ERROR		JSR	PC.ERROR
	017742	004767	037732				
3994	017746			ENDIF	\$50642:		
	017746						
3995	017746			ENDDO		BR	\$50640
	017746	000765					
	017750				\$50641:		

CVTLP

```

3996 017750          LET TSTCAS := TSTCAS + #6
      017750 062767 000006 160566          ADD      #6.TSTCAS
3997 017756          LET COUNT := COUNT - #1
      017756 005367 160506          DEC      COUNT
3998          ;UNTIL COUNT EQ #0
3999 017762          IF COUNT NE #0 THEN
      017762 005767 160502          TST      COUNT
      017766 001402          BEQ      $50643
4000 017770          INLINE <JMP $5077>
      017770 000167 176744          JMP     $5077
4001 017774          ENDIF
      017774
4002
4003
4004          ;+
4005          ; DO ONE IN-LINE CASE   CVTLPI
4006          ;-
4007 017774          CALL CLRREG
      017774 004767 036440          JSR     PC,CLRREG
4008 020000          CALL CLRDSB
      020000 004767 036670          JSR     PC,CLRDSB
4009 020004          INLINE <CVTLPI>
      020004 076177          CVTLPI
4010 020006          INLINE < .WORD      TCVLPI >
      020006 020336          .WORD   TCVLPI
4011 020010          INLINE < .WORD      TCVLPI+4 >
      020010 020342          .WORD   TCVLPI+4
4012 020012          LET CC := PSW
      020012 016767 157760 160500      MOV     PSW,CC
4013 020020          LET CC := CC CLR.BY #177760
      020020 042767 177760 160472      BIC     #177760,CC
4014 020026          IF CC NE #0 THEN
      020026 005767 160466          TST     CC
      020032 001402          BEQ     $50644
4015 020034          CALL ERROR
      020034 004767 037640          JSR     PC,ERROR
4016 020040          ENDIF
      020040
4017 020040          IF R0 NE #0 THEN
      020040 005700          TST     R0
      020042 001402          BEQ     $50645
4018 020044          CALL ERROR
      020044 004767 037630          JSR     PC,ERROR
4019 020050          ENDIF
      020050
4020 020050          IF R1 NE #0 THEN
      020050 005701          TST     R1
      020052 001402          BEQ     $50646
4021 020054          CALL ERROR
      020054 004767 037620          JSR     PC,ERROR
4022 020060          ENDIF
      020060
4023 020060          IF R2 NE #0 THEN
      020060 005702          TST     R2
      020062 001402          BEQ     $50647
4024 020064          CALL ERROR
      020064 004767 037610          JSR     PC,ERROR

```

\$50643:

\$50644:

\$50645:

\$50646:

CVTLP

4025	020070			ENDIF				
	020070						\$50647:	
4026	020070			IF R3 NE #0 THEN				TST R3
	020070	005703						BEQ \$50650
	020072	001402						
4027	020074			CALL ERROR				JSR PC.ERROR
	020074	004767	037600					
4028	020100			ENDIF			\$50650:	
	020100							
4029	020100			IF R4 NE #0 THEN				TST R4
	020100	005704						BEQ \$50651
	020102	001402						
4030	020104			CALL ERROR				JSR PC.ERROR
	020104	004767	037570					
4031	020110			ENDIF			\$50651:	
	020110							
4032	020110			IF R5 NE #0 THEN				TST R5
	020110	005705						BEQ \$50652
	020112	001402						
4033	020114			CALL ERROR				JSR PC.ERROR
	020114	004767	037560					
4034	020120			ENDIF			\$50652:	
	020120							
4035	020120			LET R0 := #DS.DST				MOV #DS.DST,R0
	020120	012700	064544					
4036	020124			IFB (R0) NE #1 THEN				CMPB (R0),#1
	020124	122027	000001					BEQ \$50653
	020130	001402						
4037	020132			CALL ERROR				JSR PC.ERROR
	020132	004767	037542					
4038	020136			ENDIF			\$50653:	
	020136							
4039	020136			REPEAT			\$50654:	
	020136							
4040	020136			IFB (R0) NE #21 THEN				CMPB (R0),#21
	020136	122027	000021					BEQ \$50655
	020142	001402						
4041	020144			CALL ERROR				JSR PC.ERROR
	020144	004767	037530					
4042	020150			ENDIF			\$50655:	
	020150							
4043	020150			UNTIL R0 EQ #DS.DST+5				CMP R0,#DS.DST+5
	020150	020027	064551					BNE \$50654
	020154	001370						
4044	020156			IFB (R0) NE #37 THEN				CMPB (R0),#37
	020156	122027	000037					BEQ \$50656
	020162	001402						
4045	020164			CALL ERROR				JSR PC.ERROR
	020164	004767	037510					
4046	020170			ENDIF			\$50656:	
	020170							
4047	020170			LET ERRFLG := #0				CLR ERRFLG
	020170	005067	160262					
4048	020174			REPEAT			\$50657:	
	020174							
4049	020174			IFB (R0) NE #0 THEN				TSTB (R0)
	020174	105720						

CVTLP

```

4050 020176 001402                                BEQ      $50660
      020200                                CALL ERROR
      020200 004767 037474
4051 020204                                ENDIF
      020204
4052 020204                                UNTIL RO EQ #DS.DST+40 OR ERRFLG NE #0
      020204 020027 064604
      020210 001403
      020212 005767 160240
      020216 001766
      020220
4053 020220                                INLINE    <JMP XCVTLN>
      020220 000167 000122                                JMP XCVTLN
4054
4055
4056
4057
4058

```

```

4059 020224 000014    TCVTLP: .WORD 12.                ;# OF TEST CASES FOR CVTLP
4060
4061 020226 060004    .WORD 60004                ;DST. DATA TYPE AND LENGTH
4062 020230 000000    .WORD 0                    ;LONG INTEGER HIGH & SIGN
4063 020232 000000    .WORD 0                    ;LONG INTEGER LOW
4064
4065 020234 060020    .WORD 60020                ;TEST CASE NUMBER 2
4066 020236 000000    .WORD 0
4067 020240 001762    .WORD 1762
4068
4069 020242 060000    .WORD 60000                ;TEST CASE NUMBER 3
4070 020244 000000    .WORD 0
4071 020246 152361    .WORD 152361
4072
4073 020250 060010    .WORD 60010                ;TEST CASE NUMBER 4
4074 020252 074377    .WORD 74377
4075 020254 171700    .WORD 171700
4076
4077 020256 060006    .WORD 60006                ;TEST CASE NUMBER 5
4078 020260 000000    .WORD 0
4079 020262 075147    .WORD 75147
4080
4081 020264 060020    .WORD 60020                ;TEST CASE NUMBER 6
4082 020266 077777    .WORD 77777
4083 020270 177777    .WORD 177777
4084
4085 020272 060015    .WORD 60015                ;TEST CASE NUMBER 7
4086 020274 100000    .WORD 100000
4087 020276 000000    .WORD 0
4088
4089 020300 060010    .WORD 60010                ;TEST CASE NUMBER 8
4090 020302 177755    .WORD 177755
4091 020304 130070    .WORD 130070
4092
4093 020306 070010    .WORD 70010                ;TEST CASE NUMBER 9
4094 020310 177755    .WORD 177755
4095 020312 130070    .WORD 130070
4096
4097 020314 060037    .WORD 60037                ;TEST CASE NUMBER 10

```

CVTLP

4098 020316 177777  
 4099 020320 177775  
 4100  
 4101 020322 060003  
 4102 020324 000000  
 4103 020326 000257  
 4104  
 4105 020330 060000  
 4106 020332 000000  
 4107 020334 000000  
 4108  
 4109  
 4110 020336  
 4111 020336 070012  
 4112 020340 064544  
 4113 020342 032707  
 4114 020344 041072  
 4115  
 4116  
 4117  
 4118

```

      .WORD 177777
      .WORD 177775
      .WORD 60003      ;TEST CASE NUMBER 11
      .WORD 0
      .WORD 257
      .WORD 60000      ;TEST CASE NUMBER 12
      .WORD 0
      .WORD 0
TCVLPI:
      .WORD 70012      ;IN-LINE TEST CASE
      .WORD DS.DST      ;DST TYPE = UNSIGNED PACKED, LENGTH = 12
      .WORD 32707      ;DST ADDRESS
      .WORD 41072      ;LONG INTEGER LOW WORD *
                        ; LONG INTEGER HIGH WORD AND SIGN
; = 1111111111 IN BCD,
; = 1.21,21,21,21,21,37 IN PACKED BYTES

```

CVTLN

```

4120 .SBTTL CVTLN
4121 ;*****
4122 ;
4123 ; CVTLN TEST
4124 ;
4125 ;*****
4126
4127 020346 ROUTINE XCVTLN
      020346
4128 020346          INLINE      <SCOPE>          XCVTLN:
      020346 000004          SCOPE
4129 020350          INLINE      <NOP>          NOP
      020350 000240          ;ITERATION COUNT
4130 020352          LET $TIMES := #20          ;GET TEST COUNT
      020352 012767 000020 160024          MOV #20,$TIMES
4131 020360          LET COUNT := TCVTLN          MOV TCVTLN,COUNT
      020360 016767 001400 160102          ;POINT TO FIRST TEST CASE
4132 020366          LET TSTCAS := #TCVTLN+2          MOV #TCVTLN+2,TSTCAS
      020366 012767 021766 160150          ;REPEAT
4133          ;REPEAT
4134 020374          INLINE      <#5057:>          #5057:
      020374
4135
4136 ;*
4137 ; GET TEST DATA FROM TABLE
4138 ;-
4139
4140 020374          LET R0 := TSTCAS
      020374 016700 160144          MOV TSTCAS,R0
4141 020400          LET DATA00 := (R0)          ;DST. DATA TYPE AND LENGTH
      020400 012067 160156          MOV (R0),DATA00
4142 020404          LET DATA01 := (R0)          ;LONG INTEGER HIGH
      020404 012067 160154          MOV (R0),DATA01
4143 020410          LET DATA02 := (R0)          ;LONG INTEGER LOW
      020410 012067 160152          MOV (R0),DATA02
4144
4145 ;*
4146 ; PUT EXPECTED RESULTS INTO DS.EXP AND CC.EXP
4147 ;-
4148
4149 020414          CALL CLRDSB          ;CLEAR DECIMAL STRING BUFFER AREA
      020414 004767 036254          JSR PC,CLRDSB
4150 020420          LET R2 := DATA01          MOV DATA01,R2
      020420 016702 160140
4151 020424          LET R3 := DATA02          MOV DATA02,R3
      020424 016703 160136          ;CONVERT LONG INTEGER TO AN INTERMEDIATE FORM
4152 020430          CALL CVTLI          JSR PC,CVTLI
      020430 004767 010650
4153 020434          LET CC.EXP := #0          CLR CC.EXP
      020434 005067 160062
4154 020440          LET DSLEN2 := DATA00 CLR.BY #177740          ;GET DST.LEN
      020440 016767 160116 160050          MOV DATA00,DSLEN2
      020446 042767 177740 160042          BIC #177740,DSLEN2
4155 020454          LET DATTYP := DATA00 CLR.BY #107777          ;GET DATA TYPE
      020454 016767 160102 160002          MOV DATA00,DATTYP
      020462 042767 107777 157774          BIC #107777,DATTYP
4156 020470          IF DSLEN2 EQ #0 THEN          ;DST.LEN = 0

```

CVTLN

020470	005767	160022				TST	DSLEN2
4157 020474	001053					BNE	\$50664
020476	052767	000004	160016	LET CC.EXP := CC.EXP SET.BY #ZBIT			
4158 020504	005767	160004		IF DSLEN1 HI #0 THEN		BIS	#ZBIT,CC.EXP
020510	101403					TST	DSLEN1
4159 020512	052767	000002	160002	LET CC.EXP := CC.EXP SET.BY #VBIT		BLOS	\$50665
020512				ENDIF		BIS	#VBIT,CC.EXP
4161 020520				IF DATTYP EQ #40000 THEN			\$50665:
020520	026727	157740	040000				
020526	001015						
4162 020530				IF SIGNO EQ #0 THEN		CMP	DATTYP,#40000
020530	005767	157714				BNE	\$50666
020534	001004						
4163 020536				LET DS.EXP :B= #53		TST	SIGNO
020536	112767	000053	044042			BNE	\$50667
4164 020544				ELSE		MOVB	#53,DS.EXP
020544	000406						
4165 020546				LET DS.EXP :B= #55		BR	\$50670
020546	112767	000055	044032				\$50667:
4166 020554				LET CC.EXP := CC.EXP SET.BY #NBIT		MOVB	#55,DS.EXP
020554	052767	000010	157740				
4167 020562				ENDIF		BIS	#NBIT,CC.EXP
020562							
4168 020562				ENDIF			\$50670:
020562							\$50666:
4169 020562				IF DATTYP EQ #50000 THEN			
020562	026727	157676	050000				
020570	001015						
4170 020572				IF SIGNO EQ #0 THEN		CMP	DATTYP,#50000
020572	005767	157652				BNE	\$50671
020576	001004						
4171 020600				LET DS.EXP-1 :B= #53		TST	SIGNO
020600	112767	000053	043777			BNE	\$50672
4172 020606				ELSE		MOVB	#53,DS.EXP 1
020606	000406						
4173 020610				LET DS.EXP-1 :B= #55		BR	\$50673
020610	112767	000055	043767				\$50672:
4174 020616				LET CC.EXP := CC.EXP SET.BY #NBIT		MOVB	#55,DS.EXP-1
020616	052767	000010	157676				
4175 020624				ENDIF		BIS	#NBIT,CC.EXP
020624							
4176 020624				ENDIF			\$50673:
020624							\$50671:
4177 020624				ENDIF			\$50664:
020624							
4178 020624				IF DSLEN2 GT #0 AND DSLEN1 EQ #0 THEN			;DST.LEN > 0, SRC.LEN = 0
020624	005767	157666				TST	DSLEN2
020630	003420					BLE	\$50674
020632	005767	157656				TST	DSLEN1
020636	001015					BNE	\$50674
4179 020640				LET R1 := #DS.EXP			



CVTLN

4180	020640	012701	064606		LET R2 := DSLEN2	MOV	#DS.EXP,R1
	020644						
4181	020644	016702	157646		WHILE R2 HI #0 DO	MOV	DSLEN2,R2
	020650						
	020650	005702				\$50675:	TST R2
4182	020652	101404			LET (R1)+ :B= #60	BLOS	\$50676
	020654	112721	000060		LET R2 := R2 - #1	MOVB	#60,(R1)+
4183	020660				ENDDO	DEC	R2
4184	020660	005302					
	020664	000772			LET CC.EXP := CC.EXP SET.BY #ZBIT	\$50676:	BR \$50675
4185	020664	052767	000004	157630	ENDIF		BIS #ZBIT,CC.EXP
4186	020672				IF DSLEN2 GT #0 AND DSLEN1 GT #0 THEN	\$50674:	;DST.LEN > 0 AND SRC.LEN > 0
4187	020672	005767	157620				TST DSLEN2
	020676	003565					BLE \$50677
	020700	005767	157610				TST DSLEN1
	020704	003562			LET R1 := #DSBUF1 + #10. - DSLEN1		BLE \$50677
4188	020706	012701	064440				MOV #DSBUF1,R1
	020712	062701	000012				ADD #10.,R1
	020716	166701	157572		LET R2 := #DS.EXP		SUB DSLEN1,R1
4189	020722	012702	064606		IF DSLEN2 GT DSLEN1 THEN		MOV #DS.EXP,R2
4190	020726	026767	157564	157560			;DST.LEN > SRC.LEN > 0
	020734	003413			LET R2 := R2 + DSLEN2 - DSLEN1		CMP DSLEN2,DSLEN1
4191	020736	066702	157554				BLE \$50700
	020742	166702	157546		LET R0 := #DS.EXP		ADD DSLEN2,R2
4192	020746	012700	064606		WHILE R0 NE R2 DO		SUB DSLEN1,R2
4193	020752						MOV #DS.EXP,R0
	020752	020002				\$50701:	CMP R0,R2
	020754	001403			LET (R0)+ :B= #60		BEQ \$50702
4194	020756	112720	000060		ENDDO		MOVB #60,(R0)+
4195	020762	000773			ENDIF	\$50702:	BR \$50701
4196	020764				IF DSLEN2 LO DSLEN1 THEN	\$50700:	;SRC.LEN > DST.LEN > 0
4197	020764	026767	157526	157522			CMP DSLEN2,DSLEN1
	020772	103007			LET R1 := R1 + DSLEN1 - DSLEN2		BHIS \$50703
4198	020774	066701	157514				ADD DSLEN1,R1
	021000	166701	157512		LET CC.EXP := CC.EXP SET.BY #VBIT		SUB DSLEN2,R1
4199	021004	052767	000002	157510	ENDIF		BIS #VBIT,CC.EXP
4200	021012						

CVTLN

4201	021012				REPEAT	\$50703:
4202	021012				LET TEMPO :B= (R1)+ SET.BY #60	\$50704:
	021012	112167	157460			;TRANSFER THE DIGITS TO DS.EXP
	021016	152767	000060	157452		MOV B (R1)+,TEMPO
4203	021024				LET (R2)+ :B= TEMPO	BIS B #60,TEMPO
	021024	116722	157446			MOV B TEMPO,(R2)+
4204	021030				UNTIL R1 EQ #DSBUF1+12	CMP R1,#DSBUF1+12
	021030	020127	064452			BNE \$50704
	021034	001366				TST DATTYP
4205	021036				IF DATTYP EQ #0 AND SIGNO EQ #-1 THEN	BNE \$50705
	021036	005767	157422			TST DATTYP
	021042	001012				BNE \$50705
	021044	026727	157400	177777		CMP SIGNO,#-1
	021052	001006				BNE \$50705
4206	021054				LET CC.EXP := CC.EXP SET.BY #NBIT	BIS #NBIT,CC.EXP
	021054	052767	000010	157440	LET R2 := R2 - #1	DEC R2
4207	021062				LET (R2) :B= (R2) SET.BY #160	*** SIGNED ZONE, NEGATIVE ***
	021062	005302				BIS #160,(R2)
4208	021064				ENDIF	\$50705:
	021064	152712	000160		IF DATTYP EQ #20000 OR DATTYP EQ #30000 THEN	*** OVERPUNCH ***
4209	021070					CMP DATTYP,#20000
	021070	026727	157370	020000		BEQ \$50706
	021076	001404				CMP DATTYP,#30000
	021100	026727	157360	030000		BNE \$50707
	021106	001026				\$50706:
	021110				IF DATTYP EQ #20000 THEN	;TRAILING
4211	021110					CMP DATTYP,#20000
	021110	026727	157350	020000		BNE \$50710
	021116	001002				DEC R2
4212	021120				LET R2 := R2 - #1	;LEADING
	021120	005302			ELSE	BR \$50711
4213	021122					\$50710:
	021122	000402			LET R2 := #DS.EXP	MOV #DS.EXP,R2
	021124					\$50711:
4214	021124				ENDIF	MOV B (R2),RO
	021124	012702	064606		LET RO :B= (R2) CLR.BY #60	BIC B #60,RO
4215	021130					;NEG. SIGN
	021130					CMP SIGNO,# 1
4216	021130				IF SIGNO EQ #-1 THEN	BNE \$50712
	021130	111200				ADD #10.,RO
	021132	142700	000060		LET RO := RO + #10.	BIS #NBIT,CC.EXP
4217	021136				LET CC.EXP := CC.EXP SET.BY #NBIT	\$50712:
	021136	026727	157306	177777	ENDIF	;GET APPROPRIATE SIGN/DIGIT FROM TAB
	021144	001005				MOV B OVPNCH(RO),(R2)
4218	021146				LET RO := RO + #10.	
	021146	062700	000012		LET CC.EXP := CC.EXP SET.BY #NBIT	
4219	021152				ENDIF	
	021152	052767	000010	157342	LET (R2) :B= OVPNCH(RO)	
4220	021160					
	021160					
4221	021160					
	021160	116012	056330			
4222	021164				ENDIF	

LE

CVTLN

4223	021164				IF DATTYP EQ #40000 OR DATTYP EQ #50000 THEN	\$50707:	
	021164	026727	157274	040000			;*** SEPARATE ***
	021172	001404					CMP     DATTYP,#40000
	021174	026727	157264	050C00			BEQ     \$50713
	021202	001023					CMP     DATTYP,#50000
	021204						BNE     \$50714
4224	021204				IF DATTYP EQ #50000 THEN	\$50713:	
	021204	026727	157254	050000			;LEADING
	021212	001003					CMP     DATTYP,#50000
4225	021214				LET R2 := #DS.EXP - #1		BNE     \$50715
	021214	012702	064606				MOV     #DS.EXP,R2
	021220	005302					DEC     R2
4226	021222				ENDIF		
4227	021222				IF SIGNO EQ #-1 THEN	\$50715:	
	021222	026727	157222	177777			CMP     SIGNO,# 1
	021230	001006					BNE     \$50716
4228	021232				LET (R2) :B= #55	;NEGATIVE	
	021232	112712	000055				MOVB   #55,(R2)
4229	021236				LET CC.EXP := CC.EXP SET.BY #NBIT		
	021236	052767	000010	157256			BIS     #NBIT,CC.EXP
4230	021244				ELSE		
	021244	000402					BR     \$50717
	021246						
4231	021246				LET (R2) :B= #53	\$50716:	
	021246	112712	000053				;POSITIVE
	021252						MOVB   #53,(R2)
4232	021252				ENDIF		
4233	021252				ENDIF	\$50717:	
4234	021252				ENDIF	\$50714:	
	021252					\$50677:	
4235							
4236							
4237					;; SET UP GPR'S AND EXPECTED GPR'S.		
4238					;-		
4239							
4240	021252				LET R0 := DATA00	;DATA TYPE AND LENGTH	
	021252	016700	157304				MOV     DATA00,R0
4241	021256				LET R1 := #DS.DST	;DST. ADR	
	021256	012701	064544				MOV     #DS.DST,R1
4242	021262				LET R2 := DATA01		
	021262	016702	157276				MOV     DATA01,R2
4243	021266				LET R3 := DATA02		
	021266	016703	157274				MOV     DATA02,R3
4244	021272				IF #1 SETIN COUNT THEN		
	021272	032767	000001	157170			BIT     #1,COUNT
	021300	001405					BEQ     \$50720
4245	021302				LET R4 := #177777		
	021302	012704	177777				MOV     #177777,R4
4246	021306				LET R5 := #177777		
	021306	012705	177777				MOV     #177777,R5
4247	021312				ELSE		
	021312	000402					BR     \$50721
	021314						
4248	021314				LET R4 := #0	\$50720:	

CVTLN

4249	021314	005004			LET R5 := #0	CLR	R4
	021316					CLR	R5
4250	021316	005005			ENDIF		
	021320						
4251	021320				LET R0.EXP := R0		
	021320	010067	157200				
4252	021324				LET R1.EXP := R1	MOV	R0,R0.EXP
	021324	010167	157176			MOV	R1,R1.EXP
4253	021330				LET R2.EXP := #0	CLR	R2.EXP
	021330	005067	157174			CLR	R3.EXP
4254	021334				LET R3.EXP := #0	CLR	R3.EXP
	021334	005067	157172			CLR	R3.EXP
4255	021340				LET R4.EXP := R4	MOV	R4,R4.EXP
	021340	010467	157170			MOV	R4,R4.EXP
4256	021344				LET R5.EXP := R5	MOV	R5,R5.EXP
	021344	010567	157166			MOV	R5,R5.EXP
4257							
4258							
4259					; DO THE CIS INSTRUCTION		
4260					; -		
4261							
4262	021350				INLINE <CVTLN>		
	021350	076057				CVTLN	
4263							
4264							
4265							
4266					; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.		
4267					; -		
4268							
4269	021352				LET CC := PSW		
	021352	016767	156420	157140		MOV	PSW,CC
4270	021360				LET CC := CC CLR.BY #177760		
	021360	042767	177760	157132		BIC	#177760,CC
4271	021366				IF CC NE CC.EXP THEN		
	021366	026767	157126	157126		CMP	CC,CC.EXP
	021374	001402				BEQ	#50722
4272	021376				CALL ERROR		
	021376	004767	036276			JSR	PC,ERROR
4273	021402				ENDIF		
	021402						
4274	021402				IF R0 NE R0.EXP THEN		
	021402	020067	157116			CMP	R0,R0.EXP
	021406	001402				BEQ	#50723
4275	021410				CALL ERROR		
	021410	004767	036264			JSR	PC,ERROR
4276	021414				ENDIF		
	021414						
4277	021414				IF R1 NE R1.EXP THEN		
	021414	020167	157106			CMP	R1,R1.EXP
	021420	001402				BEQ	#50724
4278	021422				CALL ERROR		
	021422	004767	036252			JSR	PC,ERROR
4279	021426				ENDIF		
	021426						
4280	021426				IF R2 NE R2.EXP THEN		
	021426	020267	157076			CMP	R2,R2.EXP

CVTLN

4281	021432	001402				BEQ	\$50725
	021434			CALL ERROR			
4282	021434	004767	036240			JSR	PC,ERROR
	021440			ENDIF			
4283	021440			IF R3 NE R3.EXP THEN		\$50725:	
	021440	020367	157066			CMP	R3,R3.EXP
	021444	001402				BEQ	\$50726
4284	021446			CALL ERROR			
	021446	004767	036226			JSR	PC,ERROR
4285	021452			ENDIF			
	021452			IF R4 NE R4.EXP THEN		\$50726:	
4286	021452	020467	157056			CMP	R4,R4.EXP
	021456	001402				BEQ	\$50727
4287	021460			CALL ERROR			
	021460	004767	036214			JSR	PC,ERROR
4288	021464			ENDIF			
	021464			IF R5 NE R5.EXP THEN		\$50727:	
4289	021464	020567	157046			CMP	R5,R5.EXP
	021470	001402				BEQ	\$50730
4290	021472			CALL ERROR			
	021472	004767	036202			JSR	PC,ERROR
4291	021476			ENDIF			
	021476					\$50730:	
4292							
4293							
4294				;;			
4295				;; COMPARE THE EXPECTED DECIMAL STRING BUFFER			
4296				;; WITH ACTUAL RESULTS.			
4297				;;			
4298	021476			LET R0 := #DS.DST			
	021476	012700	064544			MOV	#DS.DST,R0
4299	021502			LET R1 := #DS.EXP			
	021502	012701	064606			MOV	#DS.EXP,R1
4300	021506			LET ERRFLG := #0			
	021506	005067	156744			CLR	ERRFLG
4301	021512			WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO			
	021512					\$50731:	
	021512	020027	064604			CMP	R0,#DS.DST+40
	021516	001410				BEQ	\$50732
	021520	005767	156732			TST	ERRFLG
	021524	001005				BNE	\$50732
4302	021526			IFB (R0) NE (R1) THEN			
	021526	122021				CMPB	(R0),(R1)
	021530	001402				BEQ	\$50733
4303	021532			CALL ERROR			
	021532	004767	036142			JSR	PC,ERROR
4304	021536			ENDIF			
	021536					\$50733:	
4305	021536			ENDDO			
	021536	000765				BR	\$50731
	021540					\$50732:	
4306	021540			LET TSTCAS := TSTCAS + #6			
	021540	062767	000006	156776		ADD	#6,TSTCAS
4307	021546			LET COUNT := COUNT - #1			

CVTLN

4308	021546	005367	156716			DEC	COUNT
4309	021552			:UNTIL COUNT EQ #0			
	021552	005767	156712	IF COUNT NE #0 THEN		TST	COUNT
	021556	001402				BEQ	\$50734
4310	021560			INLINE <JMP \$5057>			
	021560	000167	176610			JMP	\$5057
4311	021564			ENDIF			
	021564						\$50734:
4312							
4313				:*			
4314				: DO ONE IN-LINE CASE - CVTLNI			
4315				:			
4316				:			
4317	021564			CALL CLRREG			
	021564	004767	034650			JSR	PC,CLRREG
4318	021570			CALL CLRDSB			
	021570	004767	035100			JSR	PC,CLRDSB
4319	021574			INLINE <CVTLNI>			
	021574	076157					CVTLNI
4320	021576			INLINE <.WORD TCVLNI>			
	021576	022076				.WORD	TCVLNI
4321	021600			INLINE <.WORD TCVLNI+4>			
	021600	022102				.WORD	TCVLNI-4
4322	021602			LET CC := PSW			
	021602	016767	156170 156710			MOV	PSW,CC
4323	021610			LET CC := CC CLR.BY #177760			
	021610	042767	177760 156702			BIC	#177760,CC
4324	021616			IF CC NE #0 THEN			
	021616	005767	156676			TST	CC
	021622	001402				BEQ	\$50735
4325	021624			CALL ERROR			
	021624	004767	036050			JSR	PC,ERROR
4326	021630			ENDIF			
	021630						\$50735:
4327	021630			IF R0 NE #0 THEN			
	021630	005700				TST	R0
	021632	001402				BEQ	\$50736
4328	021634			CALL ERROR			
	021634	004767	036040			JSR	PC,ERROR
4329	021640			ENDIF			
	021640						\$50736:
4330	021640			IF R1 NE #0 THEN			
	021640	005701				TST	R1
	021642	001402				BEQ	\$50737
4331	021644			CALL ERROR			
	021644	004767	036030			JSR	PC,ERROR
4332	021650			ENDIF			
	021650						\$50737:
4333	021650			IF R2 NE #0 THEN			
	021650	005702				TST	R2
	021652	001402				BEQ	\$50740
4334	021654			CALL ERROR			
	021654	004767	036020			JSR	PC,ERROR
4335	021660			ENDIF			
	021660						\$50740:
4336	021660			IF R3 NE #0 THEN			

CVTLN

021660	005703				TST	R3
021662	001402				BEQ	\$50741
4337	021664			CALL ERROR		
	021664	004767	036010		JSR	PC.ERROR
4338	021670			ENDIF		
	021670				\$50741:	
4339	021670			IF R4 NE #0 THEN		
	021670	005704			TST	R4
	021672	001402			BEQ	\$50742
4340	021674			CALL ERROR		
	021674	004767	036000		JSR	PC.ERROR
4341	021700			ENDIF		
	021700				\$50742:	
4342	021700			IF R5 NE #0 THEN		
	021700	005705			TST	R5
	021702	001402			BEQ	\$50743
4343	021704			CALL ERROR		
	021704	004767	035770		JSR	PC.ERROR
4344	021710			ENDIF		
	021710				\$50743:	
4345	021710			LET R0 := #DS.DST-1		
	021710	012700	064543		MOV	#DS.DST 1,R0
4346	021714			IFB (R0)+ NE #53 THEN		
	021714	122027	000053		CMPB	(R0)+,#53
	021720	001402			BEQ	\$50744
4347	021722			CALL ERROR		
	021722	004767	035752		JSR	PC.ERROR
4348	021726			ENDIF		
	021726				\$50744:	
4349	021726			LET ERRFLG := #0		
	021726	005067	156524		CLR	ERRFLG
4350	021732			REPEAT		
	021732				\$50745:	
4351	021732			IFB (R0)+ NE #61 THEN		
	021732	122027	000061		CMPB	(R0)+,#61
	021736	001402			BEQ	\$50746
4352	021740			CALL ERROR		
	021740	004767	035734		JSR	PC.ERROR
4353	021744			ENDIF		
	021744				\$50746:	
4354	021744			UNTIL R0 EQ #DS.DST+12 OR ERRFLG NE #0		
	021744	020027	064556		CMP	R0,#DS.DST+12
	021750	001403			BEQ	\$50747
	021752	005767	156500		TST	ERRFLG
	021756	001765			BEQ	\$50745
	021760				\$50747:	
4355	021760			INLINE <JMP XCVTPL>		
	021760	000167	000122		JMP	XCVTPL
4356						
4357						
4358				;;		
4359				;; TABLE OF TEST CASES FOR CVTLN		
4360				;;		
4361	021764	000014		TCVTLN: .WORD 12.		;;# OF TEST CASES FOR CVTLN
4362						
4363	021766	000005		.WORD 5		;;DST. DATA TYPE AND LENGTH
4364	021770	000000		.WORD 0		;;LONG INTEGER HIGH & SIGN

CVTLN

```

4365 021772 000000 .WORD 0 ;LONG INTEGER LOW
4366
4367 021774 050006 .WORD 50006 ;TEST CASE NUMBER 2
4368 021776 000000 .WORD 0
4369 022000 000000 .WORD 0
4370
4371 022002 020010 .WORD 20010 ;TEST CASE NUMBER 3
4372 022004 000000 .WORD 0
4373 022006 001751 .WORD 1751
4374
4375 022010 030000 .WORD 30000 ;TEST CASE NUMBER 4
4376 022012 000000 .WORD 0
4377 022014 152361 .WORD 152361
4378
4379 022016 040000 .WORD 40000 ;TEST CASE NUMBER 5
4380 022020 000000 .WORD 0
4381 022022 024137 .WORD 24137
4382
4383 022024 050003 .WORD 50003 ;TEST CASE NUMBER 6
4384 022026 000000 .WORD 0
4385 022030 075147 .WORD 75147
4386
4387 022032 020004 .WORD 20004 ;TEST CASE NUMBER 7
4388 022034 000000 .WORD 0
4389 022036 075147 .WORD 75147
4390
4391 022040 000012 .WORD 12 ;TEST CASE NUMBER 8
4392 022042 077777 .WORD 77777
4393 022044 177777 .WORD 177777
4394
4395 022046 000037 .WORD 37 ;TEST CASE NUMBER 9
4396 022050 100000 .WORD 100000
4397 022052 000000 .WORD 0
4398
4399 022054 010021 .WORD 10021 ;TEST CASE NUMBER 10
4400 022056 100000 .WORD 100000
4401 022060 000000 .WORD 0
4402
4403 022062 040012 .WORD 40012 ;TEST CASE NUMBER 11
4404 022064 074377 .WORD 74377
4405 022066 171700 .WORD 171700
4406
4407 022070 030000 .WORD 30000 ;TEST CASE NUMBER 12
4408 022072 000000 .WORD 0
4409 022074 000000 .WORD 0
4410
4411 022076 TCVLNI: ;IN-LINE TEST CASE
4412 022076 050012 .WORD 50012 ;DST TYPE = LEADING SEPARATE, LENGTH = 12
4413 022100 064544 .WORD DS.DST ;DST ADDRESS
4414 022102 032707 .WORD 32707 ;LONG INTEGER LOW WORD *
4415 022104 041072 .WORD 41072 ; LONG INTEGER HIGH WORD AND SIGN
4416
4417 ; = 1111111111 IN BCD
4418

```



CVTPL

```

4420 .SBTTL CVTPL
4421 ;*****
4422 ;
4423 ; CVTPL TEST
4424 ;
4425 ;*****
4426
4427 022106 ROUTINE XCVTPL
      022106
4428 022106          XCVTPL:
      022106 000004          INLINE      <SCOPE>          SCOPE
4429 022110          INLINE      <NOP>          NOP
      022110 000240
4430 022112          LET $TIMES := #20          ;ITERATION COUNT
      022112 012767 000020 156264          ;GET TEST COUNT          MOV      #20,$TIMES
4431 022120          LET COUNT := TCVTPL          ;POINT TO FIRST TEST CASE
      022120 016767 001006 156342          ;MOV      TCVTPL,COUNT
4432 022126          LET TSTCAS := #TCVTPL*2          MOV      #TCVTPL*2,TSTCAS
      022126 012767 023134 156410
4433 ;REPEAT
4434 022134          INLINE      <#5073:>
      022134
4435
4436 ;*
4437 ; GET TEST DATA FROM TABLE
4438 ;-
4439
4440 022134          LET R0 := TSTCAS
      022134 016700 156404          ;MOV      TSTCAS,R0
4441 022140          LET DATA00 := (R0)*          ;SRC. DATA TYPE AND LENGTH
      022140 012067 156416          ;MOV      (R0),DATA00
4442 022144          LET DATA01 := (R0)          ;SRC. POINTER
      022144 011067 156414          ;MOV      (R0),DATA01
4443
4444 ;*
4445 ; SET UP R0.EXP, R1.EXP, R4.EXP, R5.EXP, R4, AND R5.
4446 ;-
4447
4448 022150          LET R0.EXP := #0
      022150 005067 156350          CLR      R0.EXP
4449 022154          LET R1.EXP := #0
      022154 005067 156346          CLR      R1.EXP
4450 022160          IF #1 SETIN COUNT THEN
      022160 032767 000001 156302          BIT      #1,COUNT
      022166 001405          BEQ      #50752
4451 022170          LET R4 := #177777
      022170 012704 177777          MOV      #177777,R4
4452 022174          LET R5 := #177777
      022174 012705 177777          MOV      #177777,R5
4453 022200          ELSE
      022200 000402          BR      #50753
      022202
4454 022202          LET R4 := #0
      022202 005004          CLR      R4
4455 022204          LET R5 := #0
      022204 005005          CLR      R5
4456 022206          ENDIF

```

```

CVTPL
4457 022206          022206          LET R4.EXP := R4          $50753:
022206 010467 156322          MOV      R4,R4.EXP
4458 022212          022212          LET R5.EXP := R5          MOV      R5,R5.EXP
022212 010567 156320
4459
4460          ;+
4461          ; CONVERT PACKED NUMBER AND LOAD INTO R2.EXP AND R3.EXP
4462          ;-
4463
4464 022216          022216 004767 034452          CALL CLRDSB          JSR      PC,CLRDSB
022216 004767 034452          LET CC.EXP := #0          CLR      CC.EXP
4465 022222          022222 005067 156274          LET R0 := #DSBUF1          ;DST.POINTER          MOV      #DSBUF1,R0
022222 005067 156274          LET R1 := DATA01          ;SRC.POINTER          MOV      DATA01,R1
4466 022226          022226 012700 064440          LET DSLEN1 := DATA00 CLR.BY #177740          ;GET SRC.LEN          MOV      DATA00,DSLEN1
022226 012700 064440          IF DSLEN1 EQ #0 THEN          ;SRC.LEN=0          BIC      #177740,DSLEN1
4467 022232          022232 016701 156326          LET CC.EXP := CC.EXP SET.BY #ZBIT          TST      DSLEN1
022232 016701 156326          LET R2.EXP := #0          BNE      $50754
4468 022236          022236 016767 156320 156250          LET R3.EXP := #0          BIS      #ZBIT,CC.EXP
022236 016767 156320 156250          LET R3.EXP := #0          CLR      R2.EXP
4469 022252          022252 005767 156236          ELSE          CLR      R3.EXP
022252 005767 156236          BR      $50755
022252 001010
4470 022260          022260 052767 000004 156234          LET R0 := #DSBUF1 + #10. - DSLEN1          ;DEST. FOR INTERMEDIATE RESULTS          $50754:
022260 052767 000004 156234          MOV      #DSBUF1,R0
4471 022266          022266 005067 156236          ADD      #10.,R0
022266 005067 156236          SUB      DSLEN1,R0
4472 022272          022272 005067 156234          ;LOAD UP DSBUF1 BUFFER WITH INTERMEDIATE RESULTS
022272 005067 156234          CALL CLRDSB          ;CLEAR DECIMAL STRING BUFFERS
4473 022276          022276 000532          IF #1 NOTSETIN DSLEN1 THEN          JSR      PC,CLRDSB
022276 000532          LET (R1) :B= (R1) CLR.BY #360          ;CLEAR UPPER NIBBLE          BIT      #1,DSLEN1
4474 022300          022300 012700 064440          LET (R0)+ :B= (R1)+          BNE      $50756
022300 012700 064440          LET DSLEN1 := DSLEN1 - #1          BICB     #360,(R1)
4475 022304          022304 062700 000012          MOV      (R1)+,(R0)+
022304 062700 000012          ENDF          DEC      DSLEN1
4476 022310          022310 166700 156200          WHILE DSLEN1 HI #1 DO          $50756:
4477          $50757:
4478 022314          022314 004767 034354          CMP      DSLEN1,#1
022314 004767 034354          BIT      #1,DSLEN1
4479 022320          022320 032767 000001 156166          BLOS     $50760
022320 032767 000001 156166          LET (R1) :B= (R1) CLR.BY #360          ;CLEAR UPPER NIBBLE
022326 001005          LET (R0)+ :B= (R1)+
4480 022330          022330 142711 000360          LET DSLEN1 := DSLEN1 - #1
4481 022334          022334 112120          ENDF
4482 022336          022336 005367 156152          WHILE DSLEN1 HI #1 DO
4483 022342          022342
4484 022342          022342 026727 156146 000001          CMP      DSLEN1,#1
022342 026727 156146 000001          BLOS     $50760
022350 101420

```

CVTPL

4485	022352				LET (R0) :B= (R1)		
	022352	111110				MOVB	(R1).(R0)
4486	022354				INLINE <ASRB (R0)>		
	022354	106210				ASRB	(R0)
4487	022356				INLINE <ASRB (R0)>		
	022356	106210				ASRB	(R0)
4488	022360				INLINE <ASRB (R0)>		
	022360	106210				ASRB	(R0)
4489	022362				INLINE <ASRB (R0)>		
	022362	106210				ASRB	(R0)
4490	022364				LET (R0) :B= (R0) CLR.BY #360	;CLEAR	UPPER NIBBLE
	022364	142710	000360			BICB	#360.(R0)
4491	022370				LET R0 := R0 + #1		
	022370	005200				INC	R0
4492	022372				LET (R0) :B= (R1)+ CLR.BY #360		
	022372	112110				MOVB	(R1).,(R0)
	022374	142710	000360			BICB	#360.(R0)
4493	022400				LET R0 := R0 + #1		
	022400	005200				INC	R0
4494	022402				LET DSLEN1 := DSLEN1 - #2		
	022402	162767	000002	156104		SUB	#2.DSLEN1
4495	022410				ENDDO		
	022410	000754				BR	\$50757
	022412						
4496	022412				LET (R0) :B= (R1)	\$50760:	
	022412	111110				;GET THE	LSD
4497	022414				INLINE <ASRB (R0)>		
	022414	106210				MOVB	(R1).(R0)
4498	022416				INLINE <ASRB (R0)>		
	022416	106210				ASRB	(R0)
4499	022420				INLINE <ASRB (R0)>		
	022420	106210				ASRB	(R0)
4500	022422				INLINE <ASRB (R0)>		
	022422	106210				ASRB	(R0)
4501	022424				LET (R0) :B= (R0) CLR.BY #360		
	022424	142710	000360			BICB	#360.(R0)
4502							
4503					;LOAD UP R2.EXP AND R3.EXP WITH FINAL RESULT		
4504							
4505	022430				CALL CVTIL		
	022430	004767	007554			JSR	PC.CVTIL
4506	022434				LET DATTP := DATA00 CLR.BY #107777		
	022434	016767	156122	156022		MOV	DATA00,DATTP
	022442	042767	107777	156014		BIC	#107777,DATTP
4507	022450				IF DATTP EQ #60000 THEN ;SIGNED PACKED		
	022450	026727	156010	060000		CMP	DATTP,#60000
	022456	001042				BNE	\$50761
4508	022460				LET TEMP5 :B= (R1) CLR.BY #360 ;ISOLATE THE SIGN		
	022460	111167	156024			MOVB	(R1).TEMP5
	022464	142767	000360	156016		BICB	#360,TEMP5
4509	022472				IFB TEMP5 EQ #15 OR TEMP5 EQ #13 THEN		
	022472	126727	156012	000015		CMPB	TEMP5,#15
	022500	001404				BEQ	\$50762
	022502	026727	156002	000013		CMP	TEMP5,#13
	022510	001025				BNE	\$50763
	022512						
4510	022512				LET CC.EXP := CC.EXP SET.BY #NBIT	\$50762:	

CVTPL

4511	022512	052767	000010	156002					
	022520				LET R3.EXP := NEGATE R3.EXP		BIS	#NBIT.CC.EXP	
4512	022524	005467	156006		LET R2.EXP := COMP R2.EXP		NEG	R3.EXP	
	022524	00167	156000				COM	R2.EXP	
4513	022530	005767	155776		IF R3.EXP EQ #0 THEN		TST	R3.EXP	
	022534	001002					BNE	#50764	
4514	022536				LET R2.EXP := R2.EXP + #1		INC	R2.EXP	
4515	022542	005267	155766		ENDIF				
4516	022542				IF R2.EXP NE #0 OR R3.EXP NE #0 THEN				\$50764:
	022542	005767	155762				TST	R2.EXP	
	022546	001003					BNE	#50765	
	022550	005767	155756				TST	R3.EXP	
	022554	001403					BEG	#50766	
	022556								\$50765:
4517	022556				LET CC.EXP := CC.EXP SET.BY #CBIT				
	022556	052767	000001	155736			BIS	#CBIT.CC.EXP	
4518	022564				ENDIF				\$50766:
4519	022564				ENDIF				\$50763:
4520	022564				ENDIF				\$50761:
4521	022564				ENDIF				\$50755:
4522									
4523									
4524					;* ; SET UP R0 THRU R3				
4525					;-				
4526									
4527	022564				LET R0 := DATA00	;SRC.DSCR			
	022564	016700	155772				MOV	DATA00,R0	
4528	022570				LET R1 := DATA01	;SRC.DSCR			
	022570	016701	155770				MOV	DATA01,R1	
4529	022574	005002			LET R2 := #0		CLR	R2	
4530	022576	005003			LET R3 := #0		CLR	R3	
4531									
4532									
4533					;* ; DO THE CIS INSTRUCTION				
4534					;-				
4535									
4536	022600				INLINE <CVTPL>				
	022600	076073							CVTPL
4537									
4538									
4539									
4540					;* ; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.				
4541					;				
4542									
4543	022602				LET CC := PSW				
	022602	016767	155170	155710			MOV	PSW,CC	
4544	022610				LET CC := CC CLR.BY #177760				

CVTPL

4545	022610	042767	177760	155702	IF CC NE CC.EXP THEN	BIC	#177760.CC
	022616						
	022616	026767	155676	155676		CMP	CC.CC.EXP
	022624	001402				BEQ	#50767
4546	022626				CALL ERROR		
	022626	004767	035046			JSR	PC.ERROR
4547	022632				ENDIF		
	022632					#50767:	
4548	022632				IF R0 NE R0.EXP THEN		
	022632	020067	155666			CMP	R0.R0.EXP
	022636	001402				BEQ	#50770
4549	022640				CALL ERROR		
	022640	004767	035034			JSR	PC.ERROR
4550	022644				ENDIF		
	022644					#50770:	
4551	022644				IF R1 NE R1.EXP THEN		
	022644	020167	155656			CMP	R1.R1.EXP
	022650	001402				BEQ	#50771
4552	022652				CALL ERROR		
	022652	004767	035022			JSR	PC.ERROR
4553	022656				ENDIF		
	022656					#50771:	
4554	022656				IF R2 NE R2.EXP THEN		
	022656	020267	155646			CMP	R2.R2.EXP
	022662	001402				BEQ	#50772
4555	022664				CALL ERROR		
	022664	004767	035010			JSR	PC.ERROR
4556	022670				ENDIF		
	022670					#50772:	
4557	022670				IF R3 NE R3.EXP THEN		
	022670	020367	155636			CMP	R3.R3.EXP
	022674	001402				BEQ	#50773
4558	022676				CALL ERROR		
	022676	004767	034776			JSR	PC.ERROR
4559	022702				ENDIF		
	022702					#50773:	
4560	022702				IF R4 NE R4.EXP THEN		
	022702	020467	155626			CMP	R4.R4.EXP
	022706	001402				BEQ	#50774
4561	022710				CALL ERROR		
	022710	004767	034764			JSR	PC.ERROR
4562	022714				ENDIF		
	022714					#50774:	
4563	022714				IF R5 NE R5.EXP THEN		
	022714	020567	155616			CMP	R5.R5.EXP
	022720	001402				BEQ	#50775
4564	022722				CALL ERROR		
	022722	004767	034752			JSR	PC.ERROR
4565	022726				ENDIF		
	022726					#50775:	
4566	022726				LET TSTCAS := TSTCAS + #4		
4567	022726	062767	000004	155610		ADD	#4.TSTCAS
4568	022734				LET COUNT := COUNT - #1		
	022734	005367	155530			DEC	COUNT
4569	022740				;UNTIL COUNT EQ #0		
4570	022740				IF COUNT NE #0 THEN		

CVTPL

	022740	005767	155524						
	022744	001402						TST	COUNT
4571	022746			INLINE <JMP \$5073>				BEQ	\$50776
	022746	000167	177162						
4572	022752			ENDIF				JMP	\$5073
	022752								
4573								\$50776:	
4574									
4575				;-					
4576				; DO ONE INLINE CASE - CVTPLI					
4577				;-					
4578	022752			CALL CLRREG					
	022752	004767	033462					JSR	PC,CLRREG
4579	022756			CALL CLRDSB					
	022756	004767	033712					JSR	PC,CLRDSB
4580	022762			INLINE <CVTPLI>					
	022762	076173							CVTPLI
4581	022764			INLINE <.WORD TCVPLI>					
	022764	023210						.WORD	TCVPLI
4582	022766			INLINE <.WORD TCVPLI+4>					
	022766	023214						.WORD	TCVPLI+4
4583	022770			LET CC := PSW					
	022770	016767	155002	155522				MOV	PSW,CC
4584	022776			LET CC := CC CLR.BY #177760					
	022776	042767	177760	155514				BIC	#177760,CC
4585	023004			IF CC NE #0 THEN					
	023004	005767	155510					TST	CC
	023010	001402						BEQ	\$50777
4586	023012			CALL ERROR					
	023012	004767	034662					JSR	PC,ERROR
4587	023016			ENDIF					
	023016							\$50777:	
4588	023016			IF R0 NE #0 THEN					
	023016	005700						TST	R0
	023020	001402						BEQ	\$51000
4589	023022			CALL ERROR					
	023022	004767	034652					JSR	PC,ERROR
4590	023026			ENDIF					
	023026							\$51000:	
4591	023026			IF R1 NE #0 THEN					
	023026	005701						TST	R1
	023030	001402						BEQ	\$51001
4592	023032			CALL ERROR					
	023032	004767	034642					JSR	PC,ERROR
4593	023036			ENDIF					
	023036							\$51001:	
4594	023036			IF R2 NE #0 THEN					
	023036	005702						TST	R2
	023040	001402						BEQ	\$51002
4595	023042			CALL ERROR					
	023042	004767	034632					JSR	PC,ERROR
4596	023046			ENDIF					
	023046							\$51002:	
4597	023046			IF R3 NE #0 THEN					
	023046	005703						TST	R3
	023050	001402						BEQ	\$51003
4598	023052			CALL ERROR					

CVTPL

```

4599 023052 004767 034622                                JSR      PC.ERROR
      023056                                ENDIF
4600 023056                                IF R4 NE #0 THEN                                $51003:
      023056 005704                                TST      R4
      023060 001402                                BEQ      $51004
4601 023062                                CALL ERROR                                JSR      PC.ERROR
      023062 004767 034612
4602 023066                                ENDIF
      023066                                $51004:
4603 023066                                IF R5 NE #0 THEN
      023066 005705                                TST      R5
      023070 001402                                BEQ      $51005
4604 023072                                CALL ERROR                                JSR      PC.ERROR
      023072 004767 034602
4605 023076                                ENDIF
      023076                                $51005:
4606 023076                                IF TCVPLI+4 NE #32707 THEN
      023076 026727 000112 032707                                CMP      TCVPLI+4,#32707
      023104 001402                                BEQ      $51006
4607 023106                                CALL ERROR                                JSR      PC.ERROR
      023106 004767 034566
4608 023112                                ENDIF
      023112                                $51006:
4609 023112                                IF TCVPLI+6 NE #41072 THEN
      023112 026727 000100 041072                                CMP      TCVPLI+6,#41072
      023120 001402                                BEQ      $51007
4610 023122                                CALL ERROR                                JSR      PC.ERROR
      023122 004767 034552
4611 023126                                ENDIF
      023126                                $51007:
4612 023126                                INLINE <JMP XCVTNL>
      023126 000167 000076                                JMP XCVTNL

```

```

4613
4614 ;+
4615 ; TABLE OF TEST CASES FOR CVTPL
4616 ;-
4617
4618 023132 000006    TCVTPL: .WORD    6                ;# OF TEST CASES FOR CVTPL
4619
4620 023134 060000    .WORD    60000                ;SRC. DATA TYPE AND LENGTH
4621 023136 023164    .WORD    CVTPL1                ;POINTER TO PACKED DECIMAL STRING
4622
4623 023140 060004    .WORD    60004                ;TEST CASE NUMBER 2
4624 023142 023165    .WORD    CVTPL2
4625
4626 023144 060004    .WORD    60004                ;TEST CASE NUMBER 3
4627 023146 023170    .WORD    CVTPL3
4628
4629 023150 060012    .WORD    60012                ;TEST CASE NUMBER 4
4630 023152 023173    .WORD    CVTPL4
4631
4632 023154 070007    .WORD    70007                ;TEST CASE NUMBER 5
4633 023156 023201    .WORD    CVTPL5
4634
4635 023160 060005    .WORD    60005                ;TEST CASE NUMBER 6
4636 023162 023205    .WORD    CVTPL6

```

CVTPL

```

4637
4638
4639 023164      014          CVTPL1: .BYTE      014          ;PACKED DECIMAL STRING 1
4640
4641 023165      001      043      114  CVTPL2: .BYTE      001,043,114      ;PACKED DECIMAL STRING 2
4642
4643 023170      001      043      115  CVTPL3: .BYTE      001,043,115      ;PACKED DECIMAL STRING 3
4644
4645 023173      002      024      164  CVTPL4: .BYTE      002,024,164,203,144,174      ;PACKED DECIMAL STRING 4
      023176      203      144      174
4646
4647 023201      022      000      007  CVTPL5: .BYTE      022,000,007,055      ;PACKED DECIMAL STRING 5
      023204      055
4648
4649 023205      061      040      174  CVTPL6: .BYTE      061,040,174          ;PACKED DECIMAL STRING 6
4650
4651
4652          .EVEN
4653
4654 023210          TCVPLI:          ;INLINE TEST CASE
4655 023210      060012          .WORD      60012          ;SRC. DATA TYPE = SIGNED PACKED, LENGTH = 12
4656 023212      023220          .WORD      PLI          ;POINTER TO SRC. STRING
4657 023214      000000          .WORD      0          ;LONG INTEGER RESULT, LOW
4658 023216      000000          .WORD      0          ;LONG INTEGER RESULT, HI, AND SIGN
4659
4660 023220      001      021      021  PLI:      .BYTE      1,21,21,21,21,21,34      ;INLINE PACKED DECIMAL SRC. STRING
      023223      021      021      021
      023226      034
4661
4662          ; = +1111111111 IN BCD
4663
4664          .EVEN
4665

```



CVTNL

```

4667 .SBTTL CVTNL
4668 ;*****
4669 ;
4670 ; CVTNL TEST
4671 ;
4672 ;*****
4673
4674 023230 ROUTINE XCVTNL
      023230
4675 023230          XCVTNL:
      023230 000004          SCOPE
4676 023232          INLINE <SCOPE>
      023232 000240          SCOPE
4677 023234          LET $TIMES := #20          ;ITERATION COUNT          NOP
      023234 012767 000020 155142          ;GET TEST COUNT          MOV          #20,$TIMES
4678 023242          LET COUNT := TCVTNL          ;POINT TO FIRST TEST CASE          MOV          TCVTNL,COUNT
      023242 016767 001246 155220          ;POINT TO FIRST TEST CASE          MOV          #TCVTNL*2,TSTCAS
4679 023250          LET TSTCAS := #TCVTNL*2          ;POINT TO FIRST TEST CASE          MOV          #TCVTNL*2,TSTCAS
      023250 012767 024516 155266
4680 ;REPEAT
4681 023256          ;REPEAT
      023256          ;REPEAT          $5053:
4682          ;
4683          ;
4684          ; GET TEST DATA FROM TABLE
4685          ;
4686          ;
4687 023256          LET RO := TSTCAS
      023256 016700 155262          MOV          TSTCAS,RO
4688 023262          LET DATA00 := (RO)*          ;SRC. DATA TYPE AND LENGTH          MOV          (RO)*,DATA00
      023262 012067 155274          ;SRC. POINTER          MOV          (RO),DATA01
4689 023266          LET DATA01 := (RO)          ;SRC. POINTER          MOV          (RO),DATA01
      023266 011067 155272
4690
4691          ;
4692          ; SET UP RO.EXP, R1.EXP, R4.EXP, R5.EXP, R4, AND R5.
4693          ;
4694          ;
4695 023272          LET RO.EXP := #0
      023272 005067 155226          CLR          RO.EXP
4696 023276          LET R1.EXP := #0
      023276 005067 155224          CLR          R1.EXP
4697 023302          IF #1 SETIN COUNT THEN
      023302 032767 000001 155160          BIT          #1,COUNT
      023310 001405          BEQ          $51012
4698 023312          LET R4 := #177777
      023312 012704 177777          MOV          #177777,R4
4699 023316          LET R5 := #177777
      023316 012705 177777          MOV          #177777,R5
4700 023322          ELSE
      023322 000402          BR          $51013
      023324          ;$51012:
4701 023324          LET R4 := #0
      023324 005004          CLR          R4
4702 023326          LET R5 := #0
      023326 005005          CLR          R5
4703 023330          ENDIF

```

CVTNL

```

4704 023330          LET R4.EXP := R4          $51013:
023330 010467 155200          MOV      R4,R4.EXP
4705 023334          LET R5.EXP := R5          MOV      R5,R5.EXP
023334 010567 155176
4706
4707
4708          ;*
4709          ; CONVERT NUMERIC DECIMAL STRING AND LOAD INTO R2.EXP AND R3.EXP
4710          ; -
4711 023340          LET CC.EXP := #0
023340 005067 155156          CLR      CC.EXP
4712 023344          LET R0 := #DSBUF1          ;DST. POINTER
023344 012700 064440          MOV      #DSBUF1,R0
4713 023350          LET R1 := DATA01          ;SRC. POINTER
023350 016701 155210          MOV      DATA01,R1
4714 023354          LET SIGN0 := #0
023354 005067 155070          CLR      SIGN0
4715 023360          LET DSLEN1 := DATA00 CLR.BY #177740          ;GET SRC.LEN
023360 016767 155176 155126          MOV      DATA00,DSLEN1
023366 042767 177740 155120          BIC      #177740,DSLEN1
4716 023374          IF DSLEN1 EQ #0 THEN          ;SRC.LEN = 0
023374 005767 155114          TST      DSLEN1
023400 001010          BNE      $51014
4717 023402          LET CC.EXP := CC.EXP SET.BY #ZBIT          BIS      #ZBIT,CC.EXP
023402 052767 000004 155112
4718 023410          LET R2.EXP := #0          CLR      R2.EXP
023410 005067 155114
4719 023414          LET R3.EXP := #0          CLR      R3.EXP
023414 005067 155112
4720 023420          ELSE          ;SRC.LEN > 0
023420 000417          BR      $51015
023422
4721 023422          LET R0 := #DSBUF1 * #10. - DSLEN1          $51014:
023422 012700 064440          MOV      #DSBUF1,R0
023426 062700 000012          ADD      #10.,R0
023432 166700 155056          SUB      DSLEN1,R0
4722 023436          LET TEMPO := DSLEN1          MOV      DSLEN1,TEMPO
023436 016767 155052 155032
4723 023444          REPEAT
023444
4724 023444          LET (R0). := (R1).          ;XFER TO INTERMEDIATE          $51016:
023444 112120          MOV      (R1).,(R0).
4725 023446          LET TEMPO := TEMPO - #1          DEC      TEMPO
023446 005367 155024
4726 023452          UNTIL TEMPO EQ #0          TST      TEMPO
023452 005767 155020          BNE      $51016
023456 001372
4727 023460          ENDIF
023460
4728 023460          LET DATTYP := DATA00 CLR.BY #107777          $51015:
023460 016767 155076 154776          MOV      DATA00,DATTYP
023466 042767 107777 154770          BIC      #107777,DATTYP
4729
4730          ;GET THE SIGN OUT OF THE INTERMEDIATE RESULTS
4731
4732 023474          IF DSLEN1 NE #0 THEN          ;SRC.LEN > 0

```

CVTNL

023474	005767	155014						TST	DSLEN1
023500	001533							BEQ	\$51017
4733	023502					IF DATTYP EQ #0 OR DATTYP EQ #10000 THEN ;*** ZONED ***			
	023502	005767	154756					TST	DATTYP
	023506	001404						BEQ	\$51020
	023510	026727	154750	010000				CMP	DATTYP,#10000
	023516	001020						BNE	\$51021
	023520								
4734	023520					LET RO := RO - #1		\$51020:	
	023520	005300						;POINT TO LAST BYTE	
								DEC	RO
4735	023522					LET TEMPO :B= (RO) CLR.BY #17			
	023522	111067	154750					MOVB	(RO),TEMPO
	023526	142767	000017	154742				BICB	#17,TEMPO
4736	023534					IFB TEMPO EQ #160 THEN		;NEGATIVE	
	023534	126727	154736	000160				CMPB	TEMPO,#160
	023542	001006						BNE	\$51022
4737	023544					LET SIGNO := #-1			
	023544	012767	177777	154676				MOV	#-1,SIGNO
4738	023552					LET CC.EXP := CC.EXP SET.BY #NBIT			
	023552	052767	000010	154742				BIS	#NBIT,CC.EXP
4739	023560					ENDIF			
	023560								
4740	023560					ENDIF		\$51022:	
	023560								
4741	023560					IF DATTYP EQ #20000 OR DATTYP EQ #30000 THEN		\$51021:	
	023560	026727	154700	020000				;*** OVERPUNCH ***	
	023566	001404						CMP	DATTYP,#20000
	023570	026727	154670	030000				BEQ	\$51023
	023576	001044						CMP	DATTYP,#30000
	023600							BNE	\$51024
4742	023600					IF DATTYP EQ #20000 THEN		\$51023:	
	023600	026727	154660	020000				;TRAILING	
	023606	001002						CMP	DATTYP,#20000
4743	023610					LET RO := RO - #1		BNE	\$51025
	023610	005300							
4744	023612					ELSE		DEC	RO
	023612	000406						;LEADING	
	023614							BR	\$51026
4745	023614					LET RO := #DSBUF1 - #10. DSLEN1		\$51025:	
	023614	012700	064440					MOV	#DSBUF1,RO
	023620	062700	000012					ADD	#10,RO
	023624	166700	154664					SUB	DSLEN1,RO
4746	023630					ENDIF			
	023630								
4747	023630					LET TEMPO :B= (RO)		\$51026:	
	023630	111067	154642					;GET SIGN/DIGIT	
4748	023634					LET R1 := #SGNDGT		MOVB	(RO),TEMPO
	023634	012701	024406					;TABLE ADDRESS	
4749	023640					REPEAT		MOV	#SGNDGT,R1
	023640								
4750	023640					LET TEMP1 :B= (R1).		\$51027:	
	023640	112167	154634					MOVB	(R1),TEMP1
4751	023644					LET R1 := R1 - #1		;SKIP OVER ONE ENTRY	
	023644	005201						INC	R1
4752	023646					UNTIL TEMPO EQ TEMP1			
	023646	026767	154624	154624				CMP	TEMPO,TEMP1
	023654	001371						BNE	\$51027

CVTNL

```

4753 023656          LET R1 := R1 - #1          ;BACK UP TO CORRECT ENTRY
      023656 005301          ;NEGATIVE          DEC R1
4754 023660          IFB #BIT07 SETIN (R1) THEN
      023660 132711 000200          LET CC.EXP := CC.EXP SET.BY #NBIT
      023664 001406          LET SIGNO := # 1
4755 023666          ENDIF
      023666 052767 000010 154626          ;PUT CORRECT DIGIT INTO INTERMEDIATE RESULTS
4756 023674          LET (R0) :B= (R1) CLR.BY #BIT07
      023674 012767 177777 154546          MOV (R1),(R0)
4757 023702          ;LEADING SEPARATE          BICB #BIT07,(R0)
      023702          ;POINT TO SIGN          MOV # -1,SIGNO
4758 023702          ENDIF
      023702 111110          IF DATTYP EQ #50000 THEN
      023704 142710 000200          ;LEADING SEPARATE          CMP DATTYP,#50000
4759 023710          ;POINT TO SIGN          BNE $51031
      023710          ;LEADING SEPARATE          MOV DATA01,R1
4760 023710          ENDIF
      023710 026727 154550 050000          ;POINT TO SIGN          DEC R1
      023716 001003          IF DATTYP EQ #50000 OR DATTYP EQ #40000 THEN
4761 023720          ;LEADING SEPARATE          CMP DATTYP,#50000
      023720 016701 154640          ;LEADING SEPARATE          BEQ $51032
      023724 005301          ;LEADING SEPARATE          CMP DATTYP,#40000
4762 023726          ;LEADING SEPARATE          BNE $51033
      023726          ;LEADING SEPARATE          IFB (R1) EQ #55 THEN
4763 023726          ;LEADING SEPARATE          ;NEGATIVE          CMPB (R1),#55
      023726 026727 154532 050000          ;NEGATIVE          BNE $51034
      023734 001404          LET SIGNO := # -1
      023736 026727 154522 040000          LET CC.EXP := CC.EXP SET.BY #NBIT
      023744 001011          ENDIF
4764 023746          ;NEGATIVE          MOV # -1,SIGNO
      023746 121127 000055          LET (R0) :B= (R0) CLR.BY #360
      023752 001006          ;NEGATIVE          BICB #360,(R0)
4765 023754          ;NEGATIVE          ;CLEAR OUT UPPER NIBBLE OF INTERMEDIATE RESULTS
      023754 012767 177777 154466          INCR RO FROM #DSBUF1 TO #DSBUF1+37 BY #1
4766 023762          ;NEGATIVE          MOV #DSBUF1,RO
      023762 052767 000010 154532          ;NEGATIVE          BR $51035
4767 023770          ;NEGATIVE          ;NEGATIVE          INC RO
      023770          ;NEGATIVE          ;NEGATIVE          $51036:
4768 023770          ;NEGATIVE          ;NEGATIVE          $51035:
      023770          ;NEGATIVE          ;NEGATIVE          CMP RO,#DSBUF1+37
4769 023770          ;NEGATIVE          ;NEGATIVE          BGT $51037
      023770          ;NEGATIVE          ;NEGATIVE          BICB #360,(RO)
4770          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
4771          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
4772          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
4773 023770          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
      023770 012700 064440          ;NEGATIVE          ;NEGATIVE          MOV #DSBUF1,RO
      023774 000401          ;NEGATIVE          ;NEGATIVE          BR $51035
      023776          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
      023776 005200          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
      024000          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
      024000 020027 064477          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
      024004 003003          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
4774 024006          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
      024006 142710 000360          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
4775 024012          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE          ;NEGATIVE
      ENDINC

```

CVTNL

024012	000771								
024014									
4776									
4777									
4778									
4779	024014								
	024014	004767	006170						
4780	024020								
	024020	026727	154424	177777					
	024026	001014							
4781	024030								
	024030	052767	000010	154464					
4782	024036								
	024036	005767	154466						
	024042	001003							
	024044	005767	154462						
	024050	001403							
	024052								
4783	024052								
	024052	052767	000001	154442					
4784	024060								
	024060								
4785	024060								
	024060								
4786									
4787									
4788									
4789									
4790									
4791	024060								
	024060	016700	154476						
4792	024064								
	024064	016701	154474						
4793	024070								
	024070	005002							
4794	024072								
	024072	005003							
4795									
4796									
4797									
4798									
4799									
4800	024074								
	024074	076053							
4801									
4802									
4803									
4804									
4805									
4806									
4807	024076								
	024076	016767	153674	154414					
4808	024104								
	024104	042767	177760	154406					
4809	024112								
	024112	026767	154402	154402					
	024120	001402							

```

$51037: BR $51036
;LOAD UP R2.EXP AND R3.EXP WITH FINAL RESULT
CALL CVTIL
IF SIGNO EQ #-1 THEN
    LET CC.EXP := CC.EXP SET.BY #NBIT
    IF R2.EXP NE #0 OR R3.EXP NE #0 THEN
        JSR PC.CVTIL
        CMP SIGNO,#-1
        BNE $51040
        BIS #NBIT,CC.EXP
        TST R2.EXP
        BNE $51041
        TST R3.EXP
        BEQ $51042
    $51041:
        BIS #CBIT,CC.EXP
    $51042:
    $51040:
MOV DATA00,R0
MOV DATA01,R1
CLR R2
CLR R3
CVTNL
; DO THE CIS INSTRUCTION
; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.
LET CC := PSW
LET CC := CC CLR.BY #177760
IF CC NE CC.EXP THEN
    MOV PSW,CC
    BIC #177760,CC
    CMP CC,CC.EXP
    BEQ $51043

```

CVTNL

4810	024122			CALL ERROR			
	024122	004767	033552			JSR	PC,ERROR
4811	024126			ENDIF			
	024126					\$51043:	
4812	024126			IF R0 NE R0.EXP THEN			
	024126	020067	154372			CMP	R0,R0.EXP
	024132	001402				BEQ	\$51044
4813	024134			CALL ERROR			
	024134	004767	033540			JSR	PC,ERROR
4814	024140			ENDIF			
	024140					\$51044:	
4815	024140			IF R1 NE R1.EXP THEN			
	024140	020167	154362			CMP	R1,R1.EXP
	024144	001402				BEQ	\$51045
4816	024146			CALL ERROR			
	024146	004767	033526			JSR	PC,ERROR
4817	024152			ENDIF			
	024152					\$51045:	
4818	024152			IF R2 NE R2.EXP THEN			
	024152	020267	154352			CMP	R2,R2.EXP
	024156	001402				BEQ	\$51046
4819	024160			CALL ERROR			
	024160	004767	033514			JSR	PC,ERROR
4820	024164			ENDIF			
	024164					\$51046:	
4821	024164			IF R3 NE R3.EXP THEN			
	024164	020367	154342			CMP	R3,R3.EXP
	024170	001402				BEQ	\$51047
4822	024172			CALL ERROR			
	024172	004767	033502			JSR	PC,ERROR
4823	024176			ENDIF			
	024176					\$51047:	
4824	024176			IF R4 NE R4.EXP THEN			
	024176	020467	154332			CMP	R4,R4.EXP
	024202	001402				BEQ	\$51050
4825	024204			CALL ERROR			
	024204	004767	033470			JSR	PC,ERROR
4826	024210			ENDIF			
	024210					\$51050:	
4827	024210			IF R5 NE R5.EXP THEN			
	024210	020567	154322			CMP	R5,R5.EXP
	024214	001402				BEQ	\$51051
4828	024216			CALL ERROR			
	024216	004767	033456			JSR	PC,ERROR
4829	024222			ENDIF			
	024222					\$51051:	
4830							
4831	024222			LET TSTCAS := TSTCAS + #4			
	024222	062767	000004 154314			ADD	#4,TSTCAS
4832	024230			LET COUNT := COUNT - #1			
	024230	005367	154234			DEC	COUNT
4833				;UNTIL COUNT EQ #0			
4834	024234			IF COUNT NE #0 THEN			
	024234	005767	154230			TST	COUNT
	024240	001402				BEQ	\$51052
4835	024242			INLINE <JMP \$5053>			
	024242	000167	177010			JMP	\$5053

```

CVTNL
4836 024246          ENDIF
      024246
4837
4838
4839                ;*
4840                ; DO ONE INLINE CASE - CVTNLI
4841                ;*
4842 024246          CALL CLRREG
      024246 004767 032166
4843 024252          CALL CLRDSB
      024252 004767 032416
4844 024256          INLINE <CVTNLI>
      024256 076153
4845 024260          INLINE < .WORD TCVNLI >
      024260 025242
4846 024262          INLINE < .WORD TCVNLI+4 >
      024262 025246
4847 024264          LET CC := PSW
      024264 016767 153506 154226
4848 024272          LET CC := CC CLR.BY #177760
      024272 042767 177760 154220
4849 024300          IF CC NE #0 THEN
      024300 005767 154214
      024304 001402
4850 024306          CALL ERROR
      024306 004767 033366
4851 024312          ENDIF
      024312
4852 024312          IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN
      024312 005700
      024314 001004
      024316 005701
      024320 001002
      024322 005702
      024324 001402
      024326
4853 024326          CALL ERROR
      024326 004767 033346
4854 024332          ENDIF
      024332
4855 024332          IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
      024332 005703
      024334 001004
      024336 005704
      024340 001002
      024342 005705
      024344 001402
      024346
4856 024346          CALL ERROR
      024346 004767 033326
4857 024352          ENDIF
      024352
4858 024352          IF TCVNLI+4 NE #32707 THEN
      024352 026727 000670 032707
      024360 001402
4859 024362          CALL ERROR
      024362 004767 033312

```

#51052:

```

JSR PC,CLRREG
JSR PC,CLRDSB
CVTNLI
.WORD TCVNLI
.WORD TCVNLI+4
MOV PSW,CC
BIC #177760,CC
TST CC
BEQ $51053
JSR PC,ERROR

```

#51053:

```

TST R0
BNE $51054
TST R1
BNE $51054
TST R2
BEQ $51055

```

#51054:

```

JSR PC,ERROR

```

#51055:

```

TST R3
BNE $51056
TST R4
BNE $51056
TST R5
BEQ $51057

```

#51056:

```

JSR PC,ERROR

```

#51057:

```

CMP TCVNLI+4,#32707
BEQ $51060
JSR PC,ERROR

```

CVTNL

```

4860 024366          ENDIF
      024366
4861 024366          IF TCVPLI*6 NE #41072 THEN
      024366 026727 176624 041072
      024374 001402
4862 024376          CALL ERROR
      024376 004767 033276
4863 024402          ENDIF
      024402
4864 024402          INLINE <JMP XCVTPN>
      024402 000167 000656

```

```

4865
4866
4867 ;+
4868 ; TABLE FOR INTERPRETING ACCEPTABLE SIGN/DIGIT COMBINATIONS FOR OVERPUNCH
4869 ;-

```

Address	Label	Byte 1	Byte 2	Byte 3	Byte 4	Value	Sign
4870	024406	173	000	060	SGNDGT: .BYTE	173,0,60,0,133,0,7,0	:+0
	024411	000	133	000			
	024414	077	000				
4871	024416	101	001	061	.BYTE	101,1,61,1	:+1
	024421	001					
4872	024422	102	002	062	.BYTE	102,2,62,2	:+2
	024425	002					
4873	024426	103	003	063	.BYTE	103,3,63,3	:+3
	024431	003					
4874	024432	104	004	064	.BYTE	104,4,64,4	:+4
	024435	004					
4875	024436	105	005	065	.BYTE	105,5,65,5	:+5
	024441	005					
4876	024442	106	006	066	.BYTE	106,6,66,6	:+6
	024445	006					
4877	024446	107	007	067	.BYTE	107,7,67,7	:+7
	024451	007					
4878	024452	110	010	070	.BYTE	110,10,70,10	:+8
	024455	010					
4879	024456	111	011	071	.BYTE	111,11,71,11	:+9
	024461	011					
4880	024462	175	200	135	.BYTE	175,200,135,200,41,200,72,200	: 0
	024465	200	041	200			
	024470	072	200				
4881	024472	112	201		.BYTE	112,201	: 1
4882	024474	113	202		.BYTE	113,202	:-2
4883	024476	114	203		.BYTE	114,203	: 3
4884	024500	115	204		.BYTE	115,204	:-4
4885	024502	116	205		.BYTE	116,205	:-5
4886	024504	117	206		.BYTE	117,206	:-6
4887	024506	120	207		.BYTE	120,207	:-7
4888	024510	121	210		.BYTE	121,210	:-8
4889	024512	122	211		.BYTE	122,211	:-9

```

4890
4891
4892 ;+
4893 ; TABLE OF TEST CASES FOR CVTNL
4894 ;-

```

```

4895
4896 024514 000055 TCVTNL: .WORD 45. ;# OF TEST CASES FOR CVTNL
4897

```



## CVTNL

Address	Offset	Value	Label	Description
4898	024516	000000	.WORD 0	;SRC. DATA TYPE AND LENGTH
4899	024520	025003	.WORD CVTNL1	;POINTER TO SRC. STRING
4900				
4901	024522	050000	.WORD 50000	;TEST CASE NUMBER 2
4902	024524	025005	.WORD CVTNL2	
4903				
4904	024526	040004	.WORD 40004	;TEST CASE NUMBER 3
4905	024530	025007	.WORD CVTNL3	
4906				
4907	024532	050004	.WORD 50004	;TEST CASE NUMBER 4
4908	024534	025015	.WORD CVTNL4	
4909				
4910	024536	030007	.WORD 30007	;TEST CASE NUMBER 5
4911	024540	025022	.WORD CVTNL5	
4912				
4913	024542	000002	.WORD 2	;TEST CASE NUMBER 6
4914	024544	025032	.WORD CVTNL6	
4915				
4916	024546	020005	.WORD 20005	;TEST CASE NUMBER 7
4917	024550	025035	.WORD CVTNL7	
4918				
4919	024552	010005	.WORD 10005	;TEST CASE NUMBER 8
4920	024554	025043	.WORD CVTNL8	
4921				
4922	024556	040004	.WORD 40004	;TEST CASE NUMBER 9
4923	024560	025051	.WORD CVTNL9	
4924				
4925	024562	000012	.WORD 12	;TEST CASE NUMBER 10
4926	024564	025057	.WORD CVTNLA	
4927				
4928	024566	020002	.WORD 20002	;TEST CASE NUMBER 11
4929	024570	025072	.WORD CVTNLB	
4930				
4931	024572	020002	.WORD 20002	;TEST CASE NUMBER 12
4932	024574	025075	.WORD CVTNLC	
4933				
4934	024576	020002	.WORD 20002	;TEST CASE NUMBER 13
4935	024600	025100	.WORD CVTNLD	
4936				
4937	024602	020002	.WORD 20002	;TEST CASE NUMBER 14
4938	024604	025103	.WORD CVTNLE	
4939				
4940	024606	020002	.WORD 20002	;TEST CASE NUMBER 15
4941	024610	025106	.WORD CVTNLF	
4942				
4943	024612	020002	.WORD 20002	;TEST CASE NUMBER 16
4944	024614	025111	.WORD CVTNLG	
4945				
4946	024616	020002	.WORD 20002	;TEST CASE NUMBER 17
4947	024620	025114	.WORD CVTNLH	
4948				
4949	024622	020002	.WORD 20002	;TEST CASE NUMBER 18
4950	024624	025117	.WORD CVTNLI	
4951				
4952	024626	020002	.WORD 20002	;TEST CASE NUMBER 19
4953	024630	025122	.WORD CVTNLJ	
4954				

CVTNL

4955	024632	020002	.WORD	20002	;TEST CASE NUMBER 20
4956	024634	025125	.WORD	CVTNLK	
4957					
4958	024636	020002	.WORD	20002	;TEST CASE NUMBER 21
4959	024640	025130	.WORD	CVTNLL	
4960					
4961	024642	020002	.WORD	20002	;TEST CASE NUMBER 22
4962	024644	025133	.WORD	CVTNLM	
4963					
4964	024646	020002	.WORD	20002	;TEST CASE NUMBER 23
4965	024650	025136	.WORD	CVTNLN	
4966					
4967	024652	020002	.WORD	20002	;TEST CASE NUMBER 24
4968	024654	025141	.WORD	CVTNLO	
4969					
4970	024656	020002	.WORD	20002	;TEST CASE NUMBER 25
4971	024660	025144	.WORD	CVTNLP	
4972					
4973	024662	020002	.WORD	20002	;TEST CASE NUMBER 26
4974	024664	025147	.WORD	CVTNLQ	
4975					
4976	024666	020002	.WORD	20002	;TEST CASE NUMBER 27
4977	024670	025152	.WORD	CVTNLR	
4978					
4979	024672	020002	.WORD	20002	;TEST CASE NUMBER 28
4980	024674	025155	.WORD	CVTNLS	
4981					
4982	024676	020002	.WORD	20002	;TEST CASE NUMBER 29
4983	024700	025160	.WORD	CVTNLT	
4984					
4985	024702	020002	.WORD	20002	;TEST CASE NUMBER 30
4986	024704	025163	.WORD	CVTNLU	
4987					
4988	024706	020002	.WORD	20002	;TEST CASE NUMBER 31
4989	024710	025166	.WORD	CVTNLV	
4990					
4991	024712	020002	.WORD	20002	;TEST CASE NUMBER 32
4992	024714	025171	.WORD	CVTNLW	
4993					
4994	024716	020002	.WORD	20002	;TEST CASE NUMBER 33
4995	024720	025174	.WORD	CVTNLX	
4996					
4997	024722	020002	.WORD	20002	;TEST CASE NUMBER 34
4998	024724	025177	.WORD	CVTNLY	
4999					
5000	024726	020002	.WORD	20002	;TEST CASE NUMBER 35
5001	024730	025202	.WORD	CVTNLZ	
5002					
5003	024732	020002	.WORD	20002	;TEST CASE NUMBER 36
5004	024734	025205	.WORD	CVTNAA	
5005					
5006	024736	020002	.WORD	20002	;TEST CASE NUMBER 37
5007	024740	025210	.WORD	CVTNBB	
5008					
5009	024742	020002	.WORD	20002	;TEST CASE NUMBER 38
5010	024744	025213	.WORD	CVTNCC	
5011					

CVTNL

5012	024746	020002				.WORD	20002		;TEST CASE NUMBER 39
5013	024750	025216				.WORD	CVTNDD		
5014									
5015	024752	020002				.WORD	20002		;TEST CASE NUMBER 40
5016	024754	025221				.WORD	CVTNEE		
5017									
5018	024756	020002				.WORD	20002		;TEST CASE NUMBER 41
5019	024760	025224				.WORD	CVTNFF		
5020									
5021	024762	020002				.WORD	20002		;TEST CASE NUMBER 42
5022	024764	025227				.WORD	CVTNNG		
5023									
5024	024766	020002				.WORD	20002		;TEST CASE NUMBER 43
5025	024770	025232				.WORD	CVTNHH		
5026									
5027	024772	020002				.WORD	20002		;TEST CASE NUMBER 44
5028	024774	025235				.WORD	CVTNII		
5029									
5030	024776	020002				.WORD	20002		;TEST CASE NUMBER 45
5031	025000	025240				.WORD	CVTNJJ		
5032									
5033	025002	000				.BYTE			;FOR LEADING SEPARATE SIGN
5034	025003	000			CVTNL1:	.BYTE			;DECIMAL NUMERIC SRC STRING 1
5035									
5036	025004	055				.BYTE	055		
5037	025005	000			CVTNL2:	.BYTE			;DECIMAL NUMERIC SRC STRING 2
5038									
5039	025006	000				.BYTE			
5040	025007	061	060	061	CVTNL3:	.BYTE	061,060,061,060,053		;DECIMAL NUMERIC SRC STRING 3
	025012	060	053						
5041									
5042	025014	053				.BYTE	053		
5043	025015	061	060	061	CVTNL4:	.BYTE	061,060,061,065		;DECIMAL NUMERIC SRC STRING 4
	025020	065							
5044									
5045	025021	000				.BYTE			
5046	025022	112	020	000	CVTNL5:	.BYTE	112,020,000,000,000,007,002		;DECIMAL NUMERIC SRC STRING 5
	025025	000	000	007					
	025030	002							
5047									
5048	025031	000				.BYTE			
5049	025032	001	171		CVTNL6:	.BYTE	001,171		;DECIMAL NUMERIC SRC STRING 6
5050									
5051	025034	000				.BYTE			
5052	025035	003	001	002	CVTNL7:	.BYTE	003,001,002,000,107		;DECIMAL NUMERIC SRC STRING 7
	025040	000	107						
5053									
5054	025042	000				.BYTE			
5055	025043	001	000	002	CVTNL8:	.BYTE	001,000,002,000,067		;DECIMAL NUMERIC SRC STRING 8
	025046	000	067						
5056									
5057	025050	000				.BYTE			
5058	025051	061	060	061	CVTNL9:	.BYTE	061,060,061,060,053		;DECIMAL NUMERIC SRC STRING 9
	025054	060	053						
5059									
5060	025056	000				.BYTE			
5061	025057	002	001	004	CVTNLA:	.BYTE	002,001,004,007,004,010,003,006,004,067		;DECIMAL NUMERIC SRC STR A

CVTNL

Address	Label	Mode	Offset	Value	Description
025062		007	004	010	
025065		003	006	004	
025070		067			
5062					
5063	025071	000			
5064	025072	061	173		CVTNLB: .BYTE 61,173 ;DECIMAL NUMERIC SRC STRING 11
5065					
5066	025074	000			
5067	025075	061	060		CVTNLC: .BYTE 61,60 ;DECIMAL NUMERIC SRC STRING 12
5068					
5069	025077	000			
5070	025100	061	133		CVTNLD: .BYTE 61,133 ;DECIMAL NUMERIC SRC STRING 13
5071					
5072	025102	000			
5073	025103	061	077		CVTNLE: .BYTE 61,77 ;DECIMAL NUMERIC SRC STRING 14
5074					
5075	025105	000			
5076	025106	061	101		CVTNLF: .BYTE 61,101 ;DECIMAL NUMERIC SRC STRING 15
5077					
5078	025110	000			
5079	025111	061	061		CVTNLG: .BYTE 61,61 ;DECIMAL NUMERIC SRC STRING 16
5080					
5081	025113	000			
5082	025114	061	102		CVTNLH: .BYTE 61,102 ;DECIMAL NUMERIC SRC STRING 17
5083					
5084	025116	000			
5085	025117	061	062		CVTNLI: .BYTE 61,62 ;DECIMAL NUMERIC SRC STRING 18
5086					
5087	025121	000			
5088	025122	061	103		CVTNLJ: .BYTE 61,103 ;DECIMAL NUMERIC SRC STRING 19
5089					
5090	025124	000			
5091	025125	061	063		CVTNLK: .BYTE 61,63 ;DECIMAL NUMERIC SRC STRING 20
5092					
5093	025127	000			
5094	025130	061	104		CVTNLL: .BYTE 61,104 ;DECIMAL NUMERIC SRC STRING 21
5095					
5096	025132	000			
5097	025133	061	064		CVTNLM: .BYTE 61,64 ;DECIMAL NUMERIC SRC STRING 22
5098					
5099	025135	000			
5100	025136	061	105		CVTNLN: .BYTE 61,105 ;DECIMAL NUMERIC SRC STRING 23
5101					
5102	025140	000			
5103	025141	061	065		CVTNLO: .BYTE 61,65 ;DECIMAL NUMERIC SRC STRING 24
5104					
5105	025143	000			
5106	025144	061	106		CVTNLP: .BYTE 61,106 ;DECIMAL NUMERIC SRC STRING 25
5107					
5108	025146	000			
5109	025147	061	066		CVTNLQ: .BYTE 61,66 ;DECIMAL NUMERIC SRC STRING 26
5110					
5111	025151	000			
5112	025152	061	107		CVTNLR: .BYTE 61,107 ;DECIMAL NUMERIC SRC STRING 27
5113					
5114	025154	000			
5115	025155	061	067		CVTNLS: .BYTE 61,67 ;DECIMAL NUMERIC SRC STRING 28

## CVTNL

5116							
5117	025157	000					
5118	025160	061	110	CVTNLT:	.BYTE	61,110	;DECIMAL NUMERIC SRC STRING 29
5119							
5120	025162	000					
5121	025163	061	070	CVTNLU:	.BYTE	61,70	;DECIMAL NUMERIC SRC STRING 30
5122							
5123	025165	000					
5124	025166	061	111	CVTNLV:	.BYTE	61,111	;DECIMAL NUMERIC SRC STRING 31
5125							
5126	025170	000					
5127	025171	061	071	CVTNLW:	.BYTE	61,71	;DECIMAL NUMERIC SRC STRING 32
5128							
5129	025173	000					
5130	025174	061	175	CVTNLX:	.BYTE	61,175	;DECIMAL NUMERIC SRC STRING 33
5131							
5132	025176	000					
5133	025177	061	135	CVTNLY:	.BYTE	61,135	;DECIMAL NUMERIC SRC STRING 34
5134							
5135	025201	000					
5136	025202	061	041	CVTNLZ:	.BYTE	61,41	;DECIMAL NUMERIC SRC STRING 35
5137							
5138	025204	000					
5139	025205	061	072	CVTNAA:	.BYTE	61,72	;DECIMAL NUMERIC SRC STRING 36
5140							
5141	025207	000					
5142	025210	061	112	CVTNBB:	.BYTE	61,112	;DECIMAL NUMERIC SRC STRING 37
5143							
5144	025212	000					
5145	025213	061	113	CVTNCC:	.BYTE	61,113	;DECIMAL NUMERIC SRC STRING 38
5146							
5147	025215	000					
5148	025216	061	114	CVTNDD:	.BYTE	61,114	;DECIMAL NUMERIC SRC STRING 39
5149							
5150	025220	000					
5151	025221	061	115	CVTNEE:	.BYTE	61,115	;DECIMAL NUMERIC SRC STRING 40
5152							
5153	025223	000					
5154	025224	061	116	CVTNFF:	.BYTE	61,116	;DECIMAL NUMERIC SRC STRING 41
5155							
5156	025226	000					
5157	025227	061	117	CVTNNGG:	.BYTE	61,117	;DECIMAL NUMERIC SRC STRING 42
5158							
5159	025231	000					
5160	025232	061	120	CVTNHH:	.BYTE	61,120	;DECIMAL NUMERIC SRC STRING 43
5161							
5162	025234	000					
5163	025235	061	121	CVTNII:	.BYTE	61,121	;DECIMAL NUMERIC SRC STRING 44
5164							
5165	025237	000					
5166	025240	061	122	CVTNJJ:	.BYTE	61,122	;DECIMAL NUMERIC SRC STRING 45
5167							
5168					.EVEN		
5169							
5170	025242			TCVNLI:			;INLINE TEST CASE
5171	025242	000012			.WORD	000012	;DATA TYPE = SIGNED ZONED, LENGTH = 12
5172	025244	025252			.WORD	NL1	;ADDRESS OF DECIMAL SRC STRING

CVTNL

```

5173 025246 000000          .WORD 0          ;LONG INTEGER RESULT, LOW
5174 025250 000000          .WORD 0          ;LONG INTEGER RESULT, HI, AND SIGN
5175
5176 025252      061      061      061 NL1:  .BYTE 61,61,61,61,61,61,61,61,61      ;  = 1111111111 IN BCD
      025255      061      061      C61
      025260      061      061      061
      025263      061
5177
5178          .EVEN

```

CVTPN

```

5180 .SBTTL CVTPN
5181 ;*****
5182 ;
5183 ; CVTPN TEST
5184 ;
5185 ;*****
5186
5187 025264 ROUTINE XCVTPN
      025264                                     XCVTPN:
5188 025264 000004                               SCOPE
      025266                               SCOPE
5189 025266 000240                               NOP
      025270                               NOP
5190 025270 012767 000020 153106                ;ITERATION COUNT
      025270 012767 000020 153106                MOV      #20,$TIMES
5191 025276 016767 001700 153164                ;GET TEST COUNT
      025276 016767 001700 153164                MOV      TCVTPN,COUNT
5192 025304 012767 027204 153232                ;POINT TO FIRST TEST CASE
      025304 012767 027204 153232                MOV      #TCVTPN+2,TSTCAS
5193 ;REPEAT
5194 025312 025312                               ;$5054:
      025312
5195
5196 ;*
5197 ; GET TEST DATA FROM TABLE
5198 ;
5199
5200 025312 016700 153226                LET R0 := TSTCAS
      025312 016700 153226                MOV      TSTCAS,R0
5201 025316 012067 153240                LET DATA00 := (R0)
      025316 012067 153240                ;SRC DATA TYPE AND LENGTH
      025316 012067 153240                MOV      (R0),DATA00
5202 025322 012067 153236                LET DATA01 := (R0)
      025322 012067 153236                ;POINTER TO PACKED SRC STRING
      025322 012067 153236                MOV      (R0),DATA01
5203 025326 011067 153234                LET DATA02 := (R0)
      025326 011067 153234                ;DST.LEN
      025326 011067 153234                MOV      (R0),DATA02
5204
5205 ;*
5206 ; SET UP R0.EXP, R1.EXP, R3.EXP, R4.EXP, R5.EXP, R4, AND R5.
5207 ;
5208
5209 025332 005067 153166                LET R0.EXP := #0
      025332 005067 153166                CLR      R0.EXP
5210 025336 005067 153164                LET R1.EXP := #0
      025336 005067 153164                CLR      R1.EXP
5211 025342 012767 064544 153162                LET R3.EXP := #DS.DST
      025342 012767 064544 153162                MOV      #DS.DST,R3.EXP
5212 025350 032767 000001 153112                IF #1 SETIN COUNT THEN
      025350 032767 000001 153112                BIT      #1,COUNT
      025356 001405 --                            BEQ      $51064
5213 025360 --                            LET R4 := #-1
      025360 012704 177777                MOV      #-1,R4
5214 025364 --                            LET R5 := #-1
      025364 012705 177777                MOV      #-1,R5
5215 025370 --                            ELSE
      025370 000402 --                            BR      $51065
      025372 --
5216 025372 --                            LET R4 := #0
      025372 --                            $51064:

```

CVTPN

```

5217 025372 005004          LET R5 := #0          CLR R4
5218 025374 005005          ENDIF                CLR R5
5219 025376 010467 153132   LET R4.EXP := R4      $51065:
5220 025402 010567 153130   LET R5.EXP := R5      MOV R4,R4.EXP
5221 025402 010567 153130   MOV R5,R5.EXP
5222
5223
5224 ;+
5225 ; INITIALIZE THE DECIMAL STRING BUFFERS AND CC.EXP.
5226 ; -
5227 025406          CALL CLRDSB
5228 025406 004767 031262   JSR PC,CLRDSB
5229 025412 005067 153104   LET CC.EXP := #0     CLR CC.EXP
5230
5231
5232 ;+
5233 ; PUT THE SOURCE LENGTH INTO DSLEN1; PUT THE DESTINATION
5234 ; LENGTH INTO DSLEN2.
5235 ; -
5236 025416          LET DSLEN1 := DATA00 CLR.BY #70000
5237 025416 016767 153140 153070   MOV DATA00,DSLEN1
5238 025424 042767 070000 153062   BIC #70000,DSLEN1
5239 025432 016767 153130 153056   LET DSLEN2 := DATA02
5240 025432          MOV DATA02,DSLEN2
5241
5242 ;+
5243 ; LET SIGN0 REFLECT THE SIGN OF THE SOURCE STRING.
5244 ; -
5245 025440          LET R0 := DSLEN1
5246 025440 016700 153050   MOV DSLEN1,R0
5247 025444          INLINE <ASR R0>
5248 025444 006200          ASR R0
5249 025446          LET R0 := R0 + DATA01          ;POINT TO THE END OF THE STRING
5250 025446 066700 153112   ADD DATA01,R0
5251 025452          LET R1 := (R0) CLR.BY #360          ;GET THE SIGN NIBBLE
5252 025452 111001          MOVB (R0),R1
5253 025454 142701 000360   BICB #360,R1
5254 025460          IF DATA00 HIS #70000 OR DSLEN1 EQ #0 THEN
5255 025460 026727 153076 070000   CMP DATA00,#70000
5256 025466 103003          BHIS $51066
5257 025470 005767 153020   TST DSLEN1
5258 025474 001003          BNE $51067
5259 025476          LET SIGN0 := #0          $51066:
5260 025476 005067 152746   CLR SIGN0
5261 025502          ELSE
5262 025502 000411          BR $51070
5263 025504          $51067:

```



CVTPN

```

5251 025504          IF R1 EQ #14 THEN
      025504 020127 000014
      025510 001003
5252 025512          LET SIGNO := #0
      025512 005067 152732
5253 025516          ELSE
      025516 000403
      025520
      025520
5254 025520          LET SIGNO := # 1
      025520 012767 177777 152722
5255 025526          ENDIF
      025526
5256 025526          ENDIF
      025526
5257
5258
5259
5260
5261
5262
5263
5264 025526          IF DSLEN2 HIS DSLEN1 THEN          ;DST.LEN >= SRC.LEN
      025526 026767 152764 152760
      025534 103403
5265 025536          LET R0 := DSLEN1
      025536 016700 152752
5266 025542          ELSE          ;SRC.LEN > DST.LEN
      025542 000402
      025544
5267 025544          LET R0 := DSLEN2
      025544 016700 152746
5268 025550          ENDIF
      025550
5269
5270
5271
5272
5273
5274
5275
5276
5277 025550          LET R1 := #DS.EXP + DATA02 #1
      025550 012701 064606
      025554 066701 153006
      025560 005301
5278 025562          LET R2 := DSLEN1
      025562 016702 152726
5279 025566          INLINE <ASR R2>
      025566 006202
5280 025570          LET R2 := R2 + DATA01
      025570 066702 152770
5281
5282
5283
5284
5285
5286

```

```

;*
; CALCULATE THE NUMBER OF CHARACTERS TO BE MOVED INTO DS.EXP;
; USE R0.
;-

```

```

;*
; MAKE R1 POINT TO THE END OF THE DESTINATION IN DS.EXP;
; MAKE R2 POINT TO THE END OF THE SOURCE STRING IN THE
; TABLE OF TEST CASES.
;

```

```

;*
; MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE STRING
; TO THE EXPECTED DESTINATION BUFFER, IN UNPACKED FORMAT.
;

```

CVTPN

```

5287
5288 025574          LET R3 :B= (R2)
-   025574 111203
5289 025576          INLINE <ASH      #-4,R3>          MOVB   (R2),R3
    025576 072327 177774          ASH    #-4,R3
5290 025602          LET (R1) :B= R3 SET.BY #60 CLR.BY #300
    025602 110311          MOVB   R3,(R1)
    025604 152711 000060          BISB  #60,(R1)
    025610 142711 000300          BICB  #300,(R1)
5291 025614          LET R0 := R0 - #1          ;DECREMENT COUNT
    025614 005300          DEC    R0
5292
5293
5294
5295
5296 ;*
5297 ; MOVE THE REST OF THE SOURCE TO THE EXPECTED DESTINATION
5298 ; BUFFER, IN UNPACKED FORMAT, USING R0 AS A COUNTER FOR
5299 ; THE NUMBER OF CHARACTERS TO MOVE.
5300 ; -
5301 025616          WHILE R0 GT #0 DO
    025616          ;*
    025616 005700          ;*
    025620 003425          ;*
5302 025622          LET R1 := R1 - #1          ;DECREMENT DEST. POINTER
    025622 005301          DEC    R1
5303 025624          LET R2 := R2 - #1          ;DECREMENT SRC. POINTER
    025624 005302          DEC    R2
5304 025626          LET (R1) :B= (R2) CLR.BY #300 SET.BY #60
    025626 111211          MOVB   (R2),(R1)
    025630 142711 000300          BICB  #300,(R1)
    025634 152711 000060          BISB  #60,(R1)
5305 025640          LET R0 := R0 - #1          ;DECREMENT COUNT
    025640 005300          DEC    R0
5306 025642          IF R0 HI #0 THEN          ;COUNT > 0
    025642 005700          ;*
    025644 101412          ;*
5307 025646          LET R1 := R1 - #1          ;DECREMENT DEST. POINTER
    025646 005301          DEC    R1
5308 025650          LET R3 :B= (R2)
    025650 111203          MOVB   (R2),R3
5309 025652          INLINE <ASH      #-4,R3>          ASH    #-4,R3
    025652 072327 177774          LET (R1) :B= R3 CLR.BY #300 SET.BY #60
    025656 110311          MOVB   R3,(R1)
    025660 142711 000300          BICB  #300,(R1)
    025664 152711 000060          BISB  #60,(R1)
5311 025670          LET R0 := R0 - #1          ;DECREMENT COUNT
    025670 005300          DEC    R0
5312 025672          ENDIF
5313 025672          ENDDO
    025672 000751          ;*
    025674          ;*
5314
5315
5316 ;*
; FILL THE REST OF THE DESTINATION WITH LEADING ZEROES (60 S).

```

CVTPN

```

5317      :-
5318
5319 025674      WHILE R1 GT #DS.EXP DO
      025674
      025674 020127 064606
      025700 003403
5320 025702      LET -(R1) :B= #60
      025702 112741 000060
5321 025706      ENDDO
      025706 000772
      025710
5322
5323
5324      ;+
5325      ; PUT THE EXPECTED VALUE OF THE NBIT INTO CC.EXP
5326      ; -
5327
5328 025710      IF DSLEN2 HI #0 AND SIGN0 EQ #-1 THEN
      025710 005767 152602
      025714 101407
      025716 026727 152526 177777
      025724 001003
5329 025726      LET CC.EXP := CC.EXP SET.BY #NBIT
      025726 052767 000010 152566
5330 025734      ENDIF
      025734
5331
5332
5333      ;+
5334      ; PUT THE EXPECTED VALUE OF THE ZBIT INTO CC.EXP
5335      ; -
5336
5337
5338 025734      IF DSLEN1 EQ #0 OR DSLEN2 EQ #0 THEN
      025734 005767 152554
      025740 001403
      025742 005767 152550
      025746 001003
      025750
5339 025750      LET CC.EXP := CC.EXP SET.BY #ZBIT
      025750 052767 000004 152544
5340 025756      ENDIF
      025756
5341
5342
5343
5344      ;+
5345      ; PUT THE EXPECTED VALUE OF THE VBIT INTO CC.EXP
5346      ; -
5347 025756      IF DSLEN1 HI DSLEN2 THEN          ;SRC.LEN > DST.LEN
      025756 026767 152532 152532
      025764 101403
5348 025766      LET CC.EXP := CC.EXP SET.BY #VBIT
      025766 052767 000002 152526
5349 025774      ENDIF
      025774
5350

```

```

$51100:      CMP      R1,#DS.EXP
              BLE      $51101
              MOV8     #60,-(R1)
              BR       $51100
$51101:
$51102:      TST      DSLEN2
              BLOS    $51102
              CMP     SIGN0,-1
              BNE     $51102
              BIS     #NBIT,CC.EXP
$51103:      TST      DSLEN1
              BEQ     $51103
              TST     DSLEN2
              BNE     $51104
              BIS     #ZBIT,CC.EXP
$51104:
$51105:      CMP      DSLEN1,DSLEN2
              BLOS    $51105
              BIS     #VBIT,CC.EXP

```

CVTPN

```

5351
5352
5353      ;+
5354      ; SAVE THE FIRST AND LAST BYTES OF THE EXPECTED DESTINATION
5355      ; IN DSTMPO AND DSTMP1.
5356      ;
5357 025774      LET DSTMPO := DS.EXP
025774 016767 036606 152552
5358 026002      MOV      DS.EXP,DSTMPO
026002 012700 064606      MOV      #DS.EXP,R0
026006 066700 - 152554    ADD      DATA02,R0
026012 005300      DEC      R0
5359 026014      LET DSTMP1 :B= (R0)
026014 111067 152535      MOVB     (R0),DSTMP1
5360
5361
5362      ;+
5363      ; DO THE CVTPN INSTRUCTION, CONVERTING EACH PACKED SOURCE
5364      ; STRING INTO ALL SIX NUMERIC DATA TYPES. DS.EXP WILL HAVE
5365      ; TO BE ALTERED FOR EACH DATA TYPE.
5366      ;-
5367
5368      ;INCR DATTYP FROM #0 TO #50000 BY #10000
5369 026020      INLINE  <CLR      DATTYP>
026020 005067 152440      CLR      DATTYP
5370 026024      INLINE  <BR $5402>
026024 000412      BR      $5402
5371 026026      INLINE  <$5401:>
026026      $5401:
5372
5373      ;CLEAR THE DESTINATION BUFFER
5374
5375 026026      LET R0 := #DS.DST-1
026026 012700 064543      MOV      #DS.DST-1,R0
5376 026032      WHILE R0 LOS #DS.DST+37 DO
026032      $51106:
026032 020027 064603      CMP      R0,#DS.DST+37
026036 101002      BHI      $51107
5377 026040      LET (R0)+ :B= #0
026040 105020      CLRB     (R0)+
5378 026042      ENDDO
026042 000773      BR      $51106
026044      $51107:
5379
5380      ;INCREMENT DATTYP
5381
5382 026044      LET DATTYP := DATTYP + #10000
026044 062767 010000 152412      ADD      #10000,DATTYP
5383 026052      INLINE  <$5402:>
026052      $5402:
5384 026052      IF DATTYP HI #50000 THEN
026052 026727 152406 050000      CMP      DATTYP,#50000
026060 101402      BLOS     $51110
5385 026062      INLINE  <JMP  $5403>
026062 000167 000676      JMP      $5403
5386 026066      ENDIF
026066      $51110:

```

CVTPN

```

5387
5388
5389
5390 026066          ;RESTORE FIRST AND LAST BYTES OF DESTINATION
      026066 116767 152462 036512      LET DS.EXP :B= DSTMPO
5391 026074          IF DATA02 NE #0 THEN
      026074 005767 152466
      026100 001406
5392 026102          LET RO := #DS.EXP-1 * DATA02
      026102 012700 064605
      026106 066700 152454
5393 026112          LET (RO) :B= DSTMP1
      026112 116710 152437
5394 026116          ENDIF
      026116
                                     $51111:
5395
5396          ;*** SIGNED ZONED ***
5397
5398 026116          IF DATTYP EQ #0 THEN
      026116 005767 152342
      026122 001022
5399 026124          LET DS.EXP-1 :B= #0          ;RESTORE AFTER LEADING SEPARATE
      026124 105067 036455
5400 026130          IF DATA02 HI #0 THEN          ;DST.LEN > 0
      026130 005767 152432
      026134 101415
5401 026136          LET RO := #DS.EXP-1 * DATA02
      026136 012700 064605
      026142 066700 152420
5402 026146          IF #NBIT NOTSETTN CC.EXP THEN
      026146 032767 000010 152346
      026154 001003
5403 026156          LET (RO) :B= (RO) SET.BY #60
      026156 152710 000060
5404 026162          ELSE
      026162 000402
      026164
5405 026164          LET (RO) :B= (RO) SET.BY #160
      026164 152710 000160
5406 026170          ENDIF
      026170
5407 026170          ENDIF
      026170
5408 026170          ENDIF
      026170
5409
5410          ;*** UNSIGNED ZONED ***
5411
5412 026170          IF DATTYP EQ #10000 THEN
      026170 026727 152270 010000
      026176 001014
5413 026200          IF DATA02 HI #0 THEN          ;DST.LEN > 0
      026200 005767 152362
      026204 101406
5414 026206          LET RO := #DS.EXP-1 * DATA02
      026206 012700 064605
      026212 066700 152350

```

```

MOV B DSTMPO,DS.EXP
TST DATA02
BEQ $51111
MOV #DS.EXP-1,RO
ADD DATA02,RO
MOV B DSTMP1,(RO)
$51111:
TST DATTYP
BNE $51112
CLR B DS.EXP 1
TST DATA02
BLOS $51113
MOV #DS.EXP-1,RO
ADD DATA02,RO
;SIGN IS +
BIT #NBIT,CC.EXP
BNE $51114
BIS B #60,(RO)
;SIGN IS -
BR $51115
$51114:
BIS B #160,(RO)
$51115:
$51113:
$51112:
CMP DATTYP,#10000
BNE $51116
TST DATA02
BLOS $51117
MOV #DS.EXP-1,RO
ADD DATA02,RO

```

```

CVTPN
5415 026216          LET (R0) :B= (R0) SET.BY #60 ;PUT IN UNSIGNED 'SIGN"
      026216 152710 000060          BISB      #60.(R0)
5416 026222          ENDIF
      026222          $51117:
5417 026222          LET CC.EXP := CC.EXP CLR.BY #NBIT
      026222 042767 000010 152272          BIC      #NBIT,CC.EXP
5418 026230          ENDIF
      026230          $51116:
5419
5420 ;*** TRAILING OVERPUNCH ***
5421
5422 026230          IF DATTYP EQ #20000 THEN
      026230 026727 152230 020000          CMP      DATTYP,#20000
      026236 001047          BNE      $51120
5423 026240          IF DSLEN2 HI #0 AND SIGNO EQ #-1 THEN
      026240 005767 152252          TST      DSLEN2
      026244 101407          BLOS     $51121
      026246 026727 152176 177777          CMP      SIGNO,# 1
      026254 001003          BNE      $51121
5424 026256          LET CC.EXP := CC.EXP SET.BY #NBIT
      026256 052767 000010 152236          BIS      #NBIT,CC.EXP
5425 026264          ENDIF
5426 026264          IF DATA02 HI #0 THEN          ;DST.LEN > 0
      026264 005767 152276          TST      DATA02
      026270 101432          BLOS     $51122
5427 026272          IF DSLEN1 NE #0 AND SIGNO EQ #-1 THEN
      026272 005767 152216          TST      DSLEN1
      026276 001407          BEQ      $51123
      026300 026727 152144 177777          CMP      SIGNO,# 1
      026306 001003          BNE      $51123
5428 026310          LET CC.EXP := CC.EXP SET.BY #NBIT
      026310 052767 000010 152204          BIS      #NBIT,CC.EXP
5429 026316          ENDIF
5430 026316          LET R0 := #DS.EXP-1 + DATA02 ;POINT TO LAST DST BYTE
      026316 012700 064605          MOV      #DS.EXP-1,R0
      026322 066700 152240          ADD      DATA02,R0
5431 026326          LET R1 :B= DSTMP1 CLR.BY #60 ;GET LSD
      026326 116701 152223          MOVB    DSTMP1,R1
      026332 142701 000060          BICB    #60,R1
5432 026336          IF SIGNO EQ #-1 THEN
      026336 026727 152106 177777          CMP      SIGNO,# 1
      026344 001002          BNE      $51124
5433 026346          LET R1 := R1 + #10. ;FORM OVPNCH TABLE OFFSET
      026346 062701 000012          ADD      #10.,R1
5434 026352          ENDIF
5435 026352          LET (R0) :B= OVPNCH(R1)
      026352 116110 056330          $51124:
      ;GET CORRECT SIGN/DIGIT
      MOVB    OVPNCH(R1),(R0)
5436 026356          ENDIF
      $51122:
5437 026356          ENDIF
      $51120:
5438
5439 ;*** LEADING OVERPUNCH ***
5440

```

CVTPN

```

5441 026356          IF DATTYP EQ #30000 THEN
      026356 026727 152102 030000
      026364 001020
5442 026366          IF DATA02 HI #0 THEN          ;DST.LEN > 0
      026366 005767 152174
      026372 101415
5443 026374          LET R1 :B= DSTMP0 CLR.BY #60          ;GET MSD
      026374 116701 152154
      026400 142701 000060
5444 026404          IF #NBIT SETIN CC.EXP THEN
      026404 032767 000010 152110
      026412 001402
5445 026414          LET R1 := R1 + #10.          ;FORM OVPNCH TABLE OFFSET
      026414 062701 000012
5446 026420          ENDIF
      026420
5447 026420          LET DS.EXP :B= OVPNCH(R1)          ;GET CORRECT SIGN/DIGIT
      026420 116167 056330 036160
      026426          ENDIF
5448 026426          ENDIF
      026426
5449 026426          ENDIF
      026426
5450
5451          ;*** TRAILING SEPARATE ***
5452
5453 026426          IF DATTYP EQ #40000 THEN
      026426 026727 152032 040000
      026434 001015
5454 026436          LET R0 := #DS.EXP + DATA02
      026436 012700 064606
      026442 066700 152120
5455 026446          IF #NBIT NOTSETIN CC.EXP THEN          ;SIGN IS +
      026446 032767 000010 152046
      026454 001003
5456 026456          LET (RC) :B= #53
      026456 112710 000053
5457 026462          ELSE          ;SIGN IS -
      026462 000402
      026464
5458 026464          LET (R0) :B= #55
      026464 112710 000055
5459 026470          ENDIF
      026470
5460 026470          ENDIF
      026470
5461
5462          ;*** LEADING SEPARATE ***
5463
5464 026470          IF DATTYP EQ #50000 THEN
      026470 026727 151770 050000
      026476 001020
5465 026500          LET R0 := #DS.EXP + DATA02
      026500 012700 064606
      026504 066700 152056
5466 026510          LET (R0) :B= #0          ;RESTORE AFTER TRAILING SEPARATE
      026510 105010
5467 026512          IF #NBIT NOTSETIN CC.EXP THEN          ;SIGN = .

```

CVTPN

026512	032767	000010	152002			BIT	#NBIT,CC.EXP
5468	026520	001004				BNE	\$51134
	026522			LET DS.EXP-1 :B= #53			
5469	026522	112767	000053	036055		MOVB	#53,DS.EXP-1
	026530			ELSE			
	026530	000403					
	026532						
5470	026532			LET DS.EXP-1 :B= #55			\$51134:
	026532	112767	000055	036045		MOVB	#55,DS.EXP-1
5471	026540			ENDIF			
	026540						\$51135:
5472	026540			ENDIF			\$51133:
	026540						
5473							
5474							
5475				;* SET UP R2.EXP, R0, R1, R2, AND R3; DO THE INSTRUCTION.			
5476				;-			
5477							
5478	026540			LET R0 := DATA00		;SRC DATA TYPE AND LENGTH	
	026540	016700	152016			MOV	DATA00,R0
5479	026544			LET R1 := DATA01		;SRC POINTER	
	026544	016701	152014				
5480	026550			LET R2 := DATA02 + DATTYP		;DST DATA TYPE AND LENGTH	
	026550	016702	152012			MOV	DATA01,R1
	026554	066702	151704			ADD	DATA02,R2
5481	026560			LET R3 := #DS.DST		;DST POINTER	
	026560	012703	064544				
5482	026564			LET R2.EXP := R2		MOV	#DS.DST,R3
	026564	010267	151740				
5483	026570			INLINE <CVTPN>		MOV	R2,R2.EXP
	026570	076054					CVTPN
5484							
5485							
5486				;* CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.			
5487				;-			
5488							
5489							
5490	026572			LET CC := PSW			
	026572	016767	151200	151720		MOV	PSW,CC
5491	026600			LET CC := CC CLR.BY #177760			
	026600	042767	177760	151712		BIC	#177760,CC
5492	026606			IF CC NE CC.EXP THEN			
	026606	026767	151706	151706			
	026614	001402				CMP	CC,CC.EXP
5493	026616			CALL ERROR		BEQ	\$51136
	026616	004767	031056			JSR	PC.ERROR
5494	026622			ENDIF			
	026622						\$51136:
5495	026622			IF R0 NE R0.EXP THEN			
	026622	020067	151676			CMP	R0,R0.EXP
	026626	001402				BEQ	\$51137
5496	026630			CALL ERROR			
	026630	004767	031044			JSR	PC.ERROR
5497	026634			ENDIF			
	026634						\$51137:
5498	026634			IF R1 NE R1.EXP THEN			
	026634	020167	151666			CMP	R1,R1.EXP



CVTPN

5499	026640	001402			BEQ	\$51140
	026642		CALL ERROR			
5500	026642	004767	031032		JSR	PC.ERROR
	026646		ENDIF			
5501	026646				\$51140:	
	026646		IF R2 NE R2.EXP THEN			
	026646	020267	151656		CMP	R2,R2.EXP
	026652	001402			BEQ	\$51141
5502	026654		CALL ERROR			
	026654	004767	031020		JSR	PC.ERROR
5503	026660		ENDIF			
	026660				\$51141:	
5504	026660		IF R3 NE R3.EXP THEN			
	026660	020367	151646		CMP	R3,R3.EXP
	026664	001402			BEQ	\$51142
5505	026666		CALL ERROR			
	026666	004767	031006		JSR	PC.ERROR
5506	026672		ENDIF			
	026672				\$51142:	
5507	026672		IF R4 NE R4.EXP THEN			
	026672	020467	151636		CMP	R4,R4.EXP
	026676	001402			BEQ	\$51143
5508	026700		CALL ERROR			
	026700	004767	030774		JSR	PC.ERROR
5509	026704		ENDIF			
	026704				\$51143:	
5510	026704		IF R5 NE R5.EXP THEN			
	026704	020567	151626		CMP	R5,R5.EXP
	026710	001402			BEQ	\$51144
5511	026712		CALL ERROR			
	026712	004767	030762		JSR	PC.ERROR
5512	026716		ENDIF			
	026716				\$51144:	
5513						
5514			;COMPARE DS.DST WITH DS.EXP			
5515						
5516	026716		LET R0 := #DS.EXP-1			
	026716	012700	064605		MOV	#DS.EXP-1,R0
5517	026722		LET R1 := #DS.DST-1			
	026722	012701	064543		MOV	#DS.DST-1,R1
5518	026726		LET ERRFLG := #0		;CLEAR THE ERROR FLAG	
	026726	005067	151524		CLR	ERRFLG
5519	026732		WHILE R0 NE #DS.EXP+40 AND ERRFLG EQ #0 DO			
	026732				\$51145:	
	026732	020027	064646		CMP	R0,#DS.EXP+40
	026736	001410			BEQ	\$51146
	026740	005767	151512		TST	ERRFLG
	026744	001005			BNE	\$51146
5520	026746		IFB (R0) NE (R1) THEN			
	026746	122021			CMPB	(R0), (R1)
	026750	001402			BEQ	\$51147
5521	026752		CALL ERROR			
	026752	004767	030722		JSR	PC.ERROR
5522	026756		ENDIF			
	026756				\$51147:	
5523	026756		ENDDO			
	026756	000765			BR	\$51145

CVTPN

```

5524 026760 ;ENDINC $51146:
5525 026760 000167 177042 151552 151552 151552 ;ENDINC
      026760 000167 177042 ;ENDINC
5526 026764 000167 177042 151552 151552 151552 ;ENDINC
      026764 000167 177042 ;ENDINC
5527 026764 062767 000006 151552 151552 151552 ;ENDINC
5528 026764 062767 000006 151552 151552 151552 ;ENDINC
5529 026772 005367 151472 151552 151552 151552 ;ENDINC
5530 026772 005367 151472 151552 151552 151552 ;ENDINC
5531 026776 005767 151466 151552 151552 151552 ;ENDINC
5532 027002 001402 151552 151552 151552 ;ENDINC
5533 027004 000167 176302 151552 151552 151552 ;ENDINC
5534 027004 000167 176302 151552 151552 151552 ;ENDINC
5535 027010 000167 176302 151552 151552 151552 ;ENDINC
5536 027010 000167 176302 151552 151552 151552 ;ENDINC
5537 027010 000167 176302 151552 151552 151552 ;ENDINC
5538 027010 000167 176302 151552 151552 151552 ;ENDINC
5539 027010 004767 027424 151552 151552 151552 ;ENDINC
5540 027014 004767 027654 151552 151552 151552 ;ENDINC
5541 027020 076154 151552 151552 151552 ;ENDINC
5542 027022 027336 151552 151552 151552 ;ENDINC
5543 027024 027342 151552 151552 151552 ;ENDINC
5544 027026 016767 150744 151464 151552 151552 ;ENDINC
5545 027034 042767 177760 151456 151552 151552 ;ENDINC
5546 027042 026727 151452 000010 151552 151552 ;ENDINC
5547 027052 001402 151552 151552 151552 ;ENDINC
5548 027052 004767 030622 151552 151552 151552 ;ENDINC
5549 027056 005700 151552 151552 151552 ;ENDINC
5550 027060 001004 151552 151552 151552 ;ENDINC
5551 027062 005701 151552 151552 151552 ;ENDINC
5552 027064 001002 151552 151552 151552 ;ENDINC
5553 027066 005702 151552 151552 151552 ;ENDINC
5554 027070 001402 151552 151552 151552 ;ENDINC
5555 027072 004767 030602 151552 151552 151552 ;ENDINC
5556 027076 151552 151552 151552 ;ENDINC

```

```

$51146:
      JMP $5401
$5403:
      ADD #6,TSTCAS
      DEC COUNT
      TST COUNT
      BEQ $51150
      JMP $5054
$51150:
      JSR PC,CLRREG
      JSR PC,CLRDSB
      CVTPNI
      .WORD TCVPNI
      .WORD TCVPNI+4
      MOV PSW,CC
      BIC #177760,CC
      CMP CC,#10
      BEQ $51151
      JSR PC,ERROR
$51151:
      TST R0
      BNE $51152
      TST R1
      BNE $51152
      TST R2
      BEQ $51153
$51152:
      JSR PC,ERROR
$51153:

```

CVTPN

```

5552 027076          IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
      027076 005703
      027100 001004
      027102 005704
      027104 001002
      027106 005705
      027110 001402
      027112
5553 027112          CALL ERROR
      027112 004767 030562
5554 027116          ENDIF
      027116
5555 027116          IFB DS.DST NE #61 THEN
      027116 126727 035422 000061
      027124 001402
5556 027126          CALL ERROR
      027126 004767 030546
5557 027132          ENDIF
      027132
5558 027132          IFB DS.DST+1 NE #62 THEN
      027132 126727 035407 000062
      027140 001402
5559 027142          CALL ERROR
      027142 004767 030532
5560 027146          ENDIF
      027146
5561 027146          IFB DS.DST+2 NE #63 THEN
      027146 126727 035374 000063
      027154 001402
5562 027156          CALL ERROR
      027156 004767 030516
5563 027162          ENDIF
      027162
5564 027162          IFB DS.DST+3 NE #164 THEN
      027162 126727 035361 000164
      027170 001402
5565 027172          CALL ERROR
      027172 004767 030502
5566 027176          ENDIF
      027176
5567 027176          INLINE <JMP XCVTPN>
      027176 000167 000150
5568
5569
5570
5571
5572
5573 027202 000011          TCVTPN: .WORD 9.          ;# OF TEST CASES FOR CVTPN
5574
5575 027204 060000          .WORD 60000          ;SRC DATA TYPE AND LENGTH
5576 027206 027272          .WORD CVTPN1          ;POINTER TO PACKED SRC STRING
5577 027210 000000          .WORD 0          ;DST.LEN
5578
5579 027212 060000          .WORD 60000          ;TEST CASE NUMBER 2
5580 027214 027273          .WORD CVTPN2
5581 027216 000005          .WORD 5
5582

```

```

TST R3
BNE $51154
TST R4
BNE $51154
TST R5
BEQ $51155
$51154:
JSR PC.ERROR
$51155:
CMPB DS.DST,#61
BEQ $51156
JSR PC.ERROR
$51156:
CMPB DS.DST+1,#62
BEQ $51157
JSR PC.ERROR
$51157:
CMPB DS.DST+2,#63
BEQ $51160
JSR PC.ERROR
$51160:
CMPB DS.DST+3,#164
BEQ $51161
JSR PC.ERROR
$51161:
JMP XCVTPN

```

```

;+
; TABLE OF TEST CASES FOR CVTPN
;-

```

CVTPN

```

5583 027220 060004 .WORD 60004 ;TEST CASE NUMBER 3
5584 027222 027274 .WORD CVTPN3
5585 027224 000000 .WORD 0
5586
5587 027226 060025 .WORD 60025 ;TEST CASE NUMBER 4
5588 027230 027277 .WORD CVTPN4
5589 027232 000014 .WORD 14
5590
5591 027234 060007 .WORD 60007 ;TEST CASE NUMBER 5
5592 027236 027312 .WORD CVTPN5
5593 027240 000003 .WORD 3
5594
5595 027242 070007 .WORD 70007 ;TEST CASE NUMBER 6
5596 027244 027316 .WORD CVTPN6
5597 027246 000003 .WORD 3
5598
5599 027250 060004 .WORD 60004 ;TEST CASE NUMBER 7
5600 027252 027322 .WORD CVTPN7
5601 027254 000037 .WORD 37
5602
5603 027256 060012 .WORD 60012 ;TEST CASE NUMBER 8
5604 027260 027325 .WORD CVTPN8
5605 027262 000012 .WORD 12
5606
5607 027264 060005 .WORD 60005 ;TEST CASE NUMBER 9
5608 027266 027333 .WORD CVTPN9
5609 027270 000005 .WORD 5
5610
5611 027272 014 CVTPN1: .BYTE 014 ;PACKED DECIMAL STRING 1
5612
5613 027273 015 CVTPN2: .BYTE 015 ;PACKED DECIMAL STRING 2
5614
5615 027274 001 043 114 CVTPN3: .BYTE 001,043,114 ;PACKED DECIMAL STRING 3
5616
5617 027277 020 043 127 CVTPN4: .BYTE 020,043,127,206,224,022,003,000,100,000 014 ;PACKED DECIMAL STRING 4
027302 206 224 022
027305 003 000 100
027310 000 014
5618
5619 027312 022 000 007 CVTPN5: .BYTE 022,000,007,055 ;PACKED DECIMAL STRING 5
027315 055
5620
5621 027316 022 000 007 CVTPN6: .BYTE 022,000,007,055 ;PACKED DECIMAL STRING 6
027321 055
5622
5623 027322 001 043 114 CVTPN7: .BYTE 001,043,114 ;PACKED DECIMAL STRING 7
5624
5625 027325 002 024 164 CVTPN8: .BYTE 002,024,164,203,144,215 ;PACKED DECIMAL STRING 8
027330 203 144 215
5626
5627 027333 061 040 174 CVTPN9: .BYTE 061,040,174 ;PACKED DECIMAL STRING 9
5628
5629 .EVEN
5630
5631 027336 TCVPNI: ;INLINE TEST CASE
5632 027336 060004 .WORD 060004 ;SIGNED PACKED, LENGTH = 4
5633 027340 027346 .WORD PN1 ;POINTER TO PACKED SRC STRING

```



CVTNP

```

5641 .SBTTL CVTNP
5642 ;*****
5643 ;
5644 ; CVTNP TEST
5645 ;
5646 ;*****
5647 ;
5648 027352 ROUTINE XCVTNP
5649 027352 XCVTNP:
027352          INLINE <SCOPE>                                SCOPE
5650 027352 000004          INLINE <NOP>                                NOP
027354          LET $TIMES := #20                                ;ITERATION COUNT
5651 027356 012767 000020 151020          LET COUNT := TCVTNP          ;GET TEST COUNT
027356          LET TSTCAS := #TCVTNP*2                        ;POINT TO FIRST TEST CASE
5652 027364 016767 001460 151076          ;REPEAT
027364          ;REPEAT
5653 027372 012767 031052 151144          ;REPEAT
027372          ;REPEAT
5654          ;REPEAT
5655 027400          ;REPEAT
027400          ;REPEAT
5656          ;REPEAT
5657          ;REPEAT
5658          ; GET TEST DATA FROM TABLE
5659          ;
5660          ;
5661 027400          LET RO := TSTCAS
027400 016700 151140          ;SRC DATA TYPE AND LENGTH
5662 027404          LET DATA00 := (RO)•                      MOV      TSTCAS,RO
027404 012067 151152          ;POINTER TO INTERMEDIATE SRC STRING
5663 027410          LET DATA01 := (RO)•                      MOV      (RO)•,DATA00
027410 012067 151150          ;DST DATA TYPE AND LENGTH
5664 027414          LET DATA02 := (RO)•                      MOV      (RO)•,DATA01
027414 011067 151146          MOV      (RO),DATA02
5665          ;
5666          ;
5667          ;
5668          ; INITIALIZE THE DECIMAL BUFFERS AND CC.EXP.
5669          ;
5670          ;
5671 027420          CALL CLRDSB
027420 004767 027250          ;
5672 027424          LET CC.EXP := #0                          JSR      PC,CLRDSB
027424 005067 151072          CLR      CC.EXP
5673          ;
5674          ;
5675          ;
5676          ; LET SIGNO REFLECT THE SIGN OF THE SOURCE STRING.
5677          ;
5678          ;
5679 027430          LET SIGNO := @DATA01
027430 017767 151130 151012          MOV      @DATA01,SIGNO
5680          ;
5681          ;
5682          ; PUT SRC MAGNITUDE INTO DSBUF1. DSBUF1 WILL BE THE SOURCE FOR THE
5683          ; CVTNP INSTRUCTION.

```

CVTNP

```

5684
5685
5686 027436          CALL CLRDSB
      027436 004767 027232
5687 027442          LET R0 := DATA01          ;GET ADDRESS OF INTERMEDIATE SRC. STRING
      027442 016700 151116
5688 027446          LET R0 := R0 + #2          ;GET PAST THE SIGN WORD
      027446 062700 000002
5689 027452          LET R1 := #DSBUF1
      027452 012701 064440
5690 027456          LET DSLEN1 := DATA00 CLR.BY #70000 ;GET SRC.LEN
      027456 016767 151100 151030
      027464 042767 070000 151022
5691 027472          INCR R2 FROM #1 TO DSLEN1 BY #1 ;TRANSFER THE MAGNITUDE
      027472 012702 000001
      027476 000401
      027500
      027500 005202
      027502
      027502 020267 151006
      027506 003002
5692 027510          LET (R1)+ :B= (R0)+
      027510 112021
5693 027512          ENDINC
      027512 000772
      027514
5694
5695
5696
5697
5698
5699
5700 027514          LET DATTYP := DATA00 CLR.BY #107777 ;GET SRC DATA TYPE
      027514 016767 151042 150742
      027522 042767 107777 150734
5701
5702
5703
5704
5705 027530          IF DATTYP EQ #0 THEN
      027530 005767 150730
      027534 001014
5706 027536          IF DSLEN1 HI #0 THEN          ;SRC.LEN > 0
      027536 005767 150752
      027542 101411
5707 027544          LET R1 := R1 #1          ;POINT TO SIGN BYTE
      027544 005301
5708 027546          IF SIGNO EQ #0 THEN          ;SIGN IS -
      027546 005767 150676
      027552 001003
5709 027554          LET (R1) :B= (R1) SET.BY #60
      027554 152711 000060
5710 027560          ELSE          ;SIGN IS -
      027560 000402
      027562
5711 027562          LET (R1) :B= (R1) SET.BY #160
      027562 152711 000160

```

CVTNP

5712	027566				ENDIF				
	027566								\$51172:
5713	027566				ENDIF				
	027566								\$51170:
5714	027566				ENDIF				
	027566								\$51167:
5715									
5716									
5717					;*** UNSIGNED ZONED ***				
5718									
5719	027566				IF DATTYP EQ #10000 THEN				
	027566	026727	150672	010000					CMP DATTYP,#10000
	027574	001006							BNE \$51173
5720	027576				IF DSLEN1 HI #0 THEN				
	027576	005767	150712						TST DSLEN1
	027602	101403							BLOS \$51174
5721	027604				LET R1 := R1 - #1				;POINT TO SIGN BYTE
	027604	005301							DEC R1
5722	027606				LET (R1) :B= (R1) SET.BY #60				
	027606	152711	000060						BISB #60,(R1)
5723	027612				ENDIF				
	027612								\$51174:
5724	027612				ENDIF				
	027612								\$51173:
5725									
5726									
5727					;*** TRAILING OVERPUNCH ***				
5728									
5729	027612				IF DATTYP EQ #20000 THEN				
	027612	026727	150646	020000					CMP DATTYP,#20000
	027620	001015							BNE \$51175
5730	027622				IF DSLEN1 HI #0 THEN				
	027622	005767	150666						TST DSLEN1
	027626	101412							BLOS \$51176
5731	027630				LET R1 := R1 - #1				;POINT TO SIGN BYTE
	027630	005301							DEC R1
5732	027632				LET R0 := (R1)				
	027632	011100							MOV (R1),R0
5733	027634				IF SIGNO EQ #-1 THEN				
	027634	026727	150610	177777					CMP SIGNO,# 1
	027642	001002							BNE \$51177
5734	027644				LET R0 := R0 * #10.				
	027644	062700	000012						ADD #10.,R0
5735	027650				ENDIF				
	027650								\$51177:
5736	027650				LET (R1) :B= OVPNCH(R0)				;PUT IN SIGN/DIGIT COMBINATION
	027650	116011	056330						MOVB OVPNCH(R0),(R1)
5737	027654				ENDIF				
	027654								\$51176:
5738	027654				ENDIF				
	027654								\$51175:
5739									
5740									
5741					;*** LEADING OVERPUNCH ***				
5742									
5743	027654				IF DATTYP EQ #30000 THEN				
	027654	026727	150604	030000					CMP DATTYP,#30000



CVTNP

5744	027662	001016					BNE	\$51200
	027664			IF DSLEN1 HI #0 THEN			TST	DSLEN1
	027670	101413	150624				BL OS	\$51201
5745	027672			LET RO :B= DSBUF1			MOVB	DSBUF1,RO
5746	027672	116700	034542					
	027676			IF SIGN0 EQ #-1 THEN			CMP	SIGN0,-1
	027676	026727	150546	177777			BNE	\$51202
	027704	001002					ADD	#10.,RO
5747	027706			LET RO := RO + #10.				
	027706	062700	000012					
5748	027712			ENDIF				
	027712						\$51202:	
5749	027712			LET DSBUF1 :B= OVPNCH(RO)				
	027712	116067	056330	034520			MOVB	OVPNCH(RO),DSBUF1
5750	027720			ENDIF				
	027720						\$51201:	
5751	027720			ENDIF				
	027720						\$51200:	
5752								
5753								
5754				;*** TRAILING SEPARATE ***				
5755								
5756	027720			IF DATTYP EQ #40000 THEN				
	027720	026727	150540	040000			CMP	DATTYP,#40000
	027726	001010					BNE	\$51203
5757	027730			IF SIGN0 EQ #0 THEN		;SIGN IS .		
	027730	005767	150514				TST	SIGN0
	027734	001003					BNE	\$51204
5758	027736			LET (R1) :B= #53				
	027736	112711	000053				MOVB	#53,(R1)
5759	027742			ELSE		;SIGN IS		
	027742	000402					BR	\$51205
	027744						\$51204:	
5760	027744			LET (R1) :B= #55				
	027744	112711	000055				MOVB	#55,(R1)
5761	027750			ENDIF				
	027750						\$51205:	
5762	027750			ENDIF				
	027750						\$51203:	
5763								
5764								
5765				;*** LEADING SEPARATE ***				
5766								
5767	027750			IF DATTYP EQ #50000 THEN				
	027750	026727	150510	050000			CMP	DATTYP,#50000
	027756	001012					BNE	\$51206
5768	027760			IF SIGN0 EQ #0 THEN		;SIGN IS .		
	027760	005767	150464				TST	SIGN0
	027764	001004					BNE	\$51207
5769	027766			LET DSBUF1-1 :B= #53				
	027766	112767	000053	034443			MOVB	#53,DSBUF1-1
5770	027774			ELSE		;SIGN IS -		
	027774	000403					BR	\$51210
	027776						\$51207:	
5771	027776			LET DSBUF1-1 :B= #55				
	027776	112767	000055	034433			MOVB	#55,DSBUF1-1

CVTNP

```

5772 030004          ENDIF
      030004
5773 030004          ENDIF
      030004
5774
5775
5776
5777
5778
5779
5780
5781 030004          LET R2 := DATA02 CLR.BY #70000          ;GET DST.LEN
      030004 016702 150556          MOV          DATA02,R2
      030010 042702 070000          BIC          #70000,R2
5782 030014          INLINE <ASR          R2>
      030014 006202          ASR          R2
5783 030016          LET R0 := #DS.EXP + R2
      030016 012700 064606          MOV          #DS.EXP,PO
      030022 060200          ADD          R2,R0
5784 030024          LET TEMP1 := DATA02 CLR.BY #107777
      030024 016767 150536 150446          MOV          DATA02,TEMP1
      030032 042767 107777 150440          BIC          #107777,TEMP1
5785
5786
5787
5788
5789
5790
5791 030040          LET DSLEN1 := DATA00 CLR.BY #70000          ;GET SRC.LEN
      030040 016767 150516 150446          MOV          DATA00,DSLEN1
      030046 042767 070000 150440          BIC          #70000,DSLEN1
5792 030054          LET DSLEN2 := DATA02 CLR.BY #70000          ;GET DST.LEN
      030054 016767 150506 150434          MOV          DATA02,DSLEN2
      030062 042767 070000 150426          BIC          #70000,DSLEN2
5793 030070          IF DSLEN1 EQ #0 THEN          ;SRC.LEN OR DST.LEN = 0
      030070 005767 150420          TST          DSLEN1
      030074 001003          BNE          $51211
5794 030076          LET (R0) :B= #14
      030076 112710 000014          MOVB         #14,(R0)
5795 030102          ELSE          ;SRC.LEN & DST.LEN > 0
      030102 000414          BR          $51212
      030104          $51211:
5796 030104          IF @DATA01 EQ #0 OR DATTYP EQ #10000 THEN
      030104 005777 150454          TST          @DATA01
      030110 001404          BEQ          $51213
      030112 026727 150346 010000          CMP          DATTYP,#10000
      030120 001003          BNE          $51214
      030122          $51213:
5797 030122          LET (R0) :B= #14          ;SIGN IS .
      030122 112710 000014          MOVB         #14,(R0)
5798 030126          ELSE
      030126 000402          BR          $51215
      030130          $51214:
5799 030130          LET (R0) :B= #15          ;SIGN IS -
      030130 112710 000015          MOVB         #15,(R0)
5800 030134          ENDIF
      030134          $51215:

```

CVTNP

```

5801 030134          ENDIF
      030134
5802 030134          IF TEMP1 EQ #70000 THEN          $51212:
      030134 026727 150340 070000
      030142 001002
5803 030144          LET (R0) :B= #17
      030144 112710 000017
5804 030150          ENDIF
      030150
5805
5806
5807
5808 ;*
5809 ; PUT THE EXPECTED VALUE OF THE NBIT INTO CC.EXP
5810 ; -
5811 030150          IF DATYP NE #10000 AND TEMP1 NE #70000 THEN
      030150 026727 150310 010000
      030156 001415
      030160 026727 150314 070000
      030166 001411
5812 030170          IFB (R0) EQ #15 AND DSLEN2 GT #0 THEN ;SIGN NIBBLE IN DS.EXP IS NEG.
      030170 121027 000015
      030174 001006
      030176 005767 150314
      030202 003403
5813 030204          LET CC.EXP := CC.EXP SET.BY #NBIT
      030204 052767 000010 150310
5814 030212          ENDIF
      030212
5815 030212          ENDIF
      030212
5816
5817
5818 ;*
5819 ; PUT THE EXPECTED VALUE OF THE ZBIT INTO CC.EXP
5820 ; -
5821
5822 030212          IF DSLEN1 EQ #0 OR DSLEN2 EQ #0 THEN ;SRC.LEN OR DST.LEN = 0
      030212 005767 150276
      030216 001403
      030220 005767 150272
      030224 001003
      030226
5823 030226          LET CC.EXP := CC.EXP SET.BY #ZBIT
      030226 052767 000004 150266
5824 030234          ENDIF
      030234
5825
5826
5827 ;*
5828 ; PUT THE EXPECTED VALUE OF THE VBIT INTO CC.EXP
5829 ; -
5830
5831 030234          IF DSLEN1 HI DSLEN2 THEN ;SRC.LEN > DST.LEN
      030234 026767 150254 150254
      030242 101403
5832 030244          LET CC.EXP := CC.EXP SET.BY #VBIT

```

CVTNP

```

5833 030244 052767 000002 150250          ENDIF          BIS          #VBIT,CC.EXP
      030252
      030252                                $51223:
5834
5835
5836
5837          ;+
5838          ; CALCULATE THE NUMBER OF CHARACTERS TO BE MOVED INTO DS.EXP;
5839          ; USE R0.
5840          ;-
5841 030252          IF DSLEN2 HIS DSLEN1 THEN          ;DST.LEN >= SRC.LEN
      030252 026767 150240 150234          ;DST.LEN >= SRC.LEN          CMP          DSLEN2,DSLEN1
      030260 103403          ;DST.LEN >= SRC.LEN          BLO          $51224
5842 030262          LET R0 := DSLEN1
      030262 016700 150226          ;DST.LEN >= SRC.LEN          MOV          DSLEN1,R0
5843 030266          ELSE          ;SRC.LEN > DST.LEN
      030266 000402          ;SRC.LEN > DST.LEN          BR          $51225
      030270          ;SRC.LEN > DST.LEN          $51224:
5844 030270          LET R0 := DSLEN2
      030270 016700 150222          ;SRC.LEN > DST.LEN          MOV          DSLEN2,R0
5845 030274          ENDIF
      030274                                $51225:
5846
5847
5848
5849          ;+
5850          ; MAKE R1 POINT TO THE END OF THE DESTINATION IN DS.EXP; MAKE
5851          ; R2 POINT TO THE END OF THE SOURCE IN THE TABLE OF TEST CASES.
5852          ;-
5853 030274          LET R1 := DSLEN2
      030274 016701 150216          ;MAKE R1 POINT TO THE END OF THE DESTINATION IN DS.EXP; MAKE
5854 030300          INLINE <ASR          R1>          MOV          DSLEN2,R1
      030300 006201          ;MAKE R1 POINT TO THE END OF THE SOURCE IN THE TABLE OF TEST CASES.
5855 030302          LET R1 := R1 + #DS.EXP          ASR          R1
      030302 062701 064606          ;MAKE R1 POINT TO THE END OF THE SOURCE IN THE TABLE OF TEST CASES.
5856 030306          LET R2 := DATA01 + DSLEN1 + #1          ADD          #DS.EXP,R1
      030306 016702 150252          ;MAKE R1 POINT TO THE END OF THE SOURCE IN THE TABLE OF TEST CASES.
      030312 066702 150176          ;MAKE R1 POINT TO THE END OF THE SOURCE IN THE TABLE OF TEST CASES.
      030316 005202          ;MAKE R1 POINT TO THE END OF THE SOURCE IN THE TABLE OF TEST CASES.
      ;MAKE R1 POINT TO THE END OF THE SOURCE IN THE TABLE OF TEST CASES.
5857
5858
5859          ;+
5860          ; MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
5861          ; EXPECTED DESTINATION BUFFER, IN PACKED FORMAT.
5862          ;-
5863
5864          IF R0 GT #0 THEN
      030320          ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
      030320 005700          ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
      030322 003405          ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
5865 030324          LET R3 := (R2)          TST          R0
      030324 111203          ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
      ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
5866 030326          INLINE <ASH          #4,R3>          BLE          $51226
      030326 072327 000004          ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
      ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
5867 030332          LET (R1) := (R1) SET.BY R3          MOVB          (R2),R3
      030332 150311          ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
      ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
5868 030334          LET R0 := R0 - #1          ASH          #4,R3
      030334 005300          ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
      ;MOVE THE LEAST SIGNIFICANT BCD FROM THE SOURCE TO THE
          ;DECREMENT COUNT          BISB          R3,(R1)
          ;DECREMENT COUNT          DEC          R0

```

CVTNP

```

5869 030336          ENDIF
      030336          $51226:
5870
5871
5872
5873          ;*
5874          ; MOVE THE REST OF THE SOURCE TO THE EXPECTED DESTINATION
5875          ; BUFFER, IN PACKED FORMAT, USING R0 AS A COUNTER FOR THE NUMBER
5876          ; OF CHARACTERS TO MOVE.
5877          ;-
5878 030336          WHILE R0 GT #0 DO
      030336
      030336          005700          $51227:
      030340          003417          TST          R0
                                          BLE          $51230
5879
5880 030342          LET R1 := R1 - #1          ;DECREMENT DEST. POINTER
      030342          005301          DEC          R1
5881 030344          LET R2 := R2 - #1          ;DECREMENT SRC. POINTER
      030344          005302          DEC          R2
5882 030346          LET (R1) :B= (R2) CLR.BY #360 ;PUT BCD INTO LOWER NIBBLE
      030346          111211          MOVB         (R2),(R1)
      030350          142711          000360        BICB         #360,(R1)
5883 030354          LET R0 := R0 - #1          ;DECREMENT COUNT
      030354          005300          DEC          R0
5884 030356          IF R0 HI #0 THEN
      030356          005700          TST          R0
      030360          101406          BLOS         $51231
5885 030362          LET R2 := R2 - #1          ;DECREMENT SRC. POINTER
      030362          005302          DEC          R2
5886 030364          LET R3 :B= (R2)
      030364          111203          MOVB         (R2),R3
5887 030366          INLINE <ASH          #4,R3>
      030366          072327          000004        ASH          #4,R3
5888 030372          LET (R1) :B= (R1) SET.BY R3 ;PUT BCD INTO UPPER NIBBLE
      030372          150311          BISB         R3,(R1)
5889 030374          LET R0 := R0 - #1          ;DECREMENT COUNT
      030374          005300          DEC          R0
5890 030376          ENDIF
5891 030376          ENDDO          $51231:
      030376          000757          BR          $51227
      030400          $51230:
5892
5893
5894          ;*
5895          ; SET UP R0-R5 AND R0.EXP-R5.EXP; THEN DO THE CVTNP INSTRUCTION.
5896          ;-
5897
5898 030400          LET R0 := DATA00          ;SRC DATA TYPE AND LENGTH
      030400          016700          150156        MOV          DATA00,R0
5899 030404          LET R1 := #DSBUF1          ;SRC ADDRESS
      030404          012701          064440        MOV          #DSBUF1,R1
5900 030410          LET R2 := DATA02          ;DST DATA TYPE AND LENGTH
      030410          016702          150152        MOV          DATA02,R2
5901 030414          LET R3 := #DS.DST          ;DST ADDRESS
      030414          012703          064544        MOV          #DS.DST,R3
5902 030420          LET R4 := #0

```



CVTNP

5931	030536			CALL ERROR			
	030556	004767	027116			JSR	PC.ERROR
5932	030562			ENDIF			
	030562					\$51236:	
5933	030562			IF R4 NE R4.EXP THEN			
	030562	020467	147746			CMP	R4,R4.EXP
	030566	001402				BEQ	\$51237
5934	030570			CALL ERROR			
	030570	004767	027104			JSR	PC.ERROR
5935	030574			ENDIF			
	030574					\$51237:	
5936	030574			IF R5 NE R5.EXP THEN			
	030574	020567	147736			CMP	R5,R5.EXP
	030600	001402				BEQ	\$51240
5937	030602			CALL ERROR			
	030602	004767	027072			JSR	PC.ERROR
5938	030606			ENDIF			
	030606					\$51240:	
5939							
5940				;COMPARE DS.DST WITH DS.EXP			
5941							
5942	030606			LET ERRFLG := #0			
	030606	005067	147644			CLR	ERRFLG
5943	030612			LET R0 := #DS.DST			
	030612	012700	064544			MOV	#DS.DST,R0
5944	030616			LET R1 := #DS.EXP			
	030616	012701	064606			MOV	#DS.EXP,R1
5945	030622			WHILE ERRFLG EQ #0 AND R0 LO #DS.DST+40 DO			
	030622					\$51241:	
	030622	005767	147630			TST	ERRFLG
	030626	001010				BNE	\$51242
	030630	020027	064604			CMP	R0,#DS.DST+40
	030634	103005				BHIS	\$51242
5946	030636			IFB (R0) NE (R1) THEN			
	030636	122021				CMPB	(R0),(R1)
	030640	001402				BEQ	\$51243
5947	030642			CALL ERROR			
	030642	004767	027032			JCR	PC.ERROR
5948	030646			ENDIF			
	030646					\$51243:	
5949	030646			ENDDO			
	030646	000765				BR	\$51241
	030650					\$51242:	
5950	030650			LET TSTCAS := TSTCAS + #6			
	030650	062767	000006	147666		ADD	#6,TSTCAS
5951	030656			LET COUNT := COUNT - #1			
	030656	005367	147606			DEC	COUNT
5952				;UNTIL COUNT EQ #0			
5953	030662			IF COUNT NE #0 THEN			
	030662	005767	147602			TST	COUNT
	030666	001402				BEQ	\$51244
5954	030670			INLINE <JMP \$5055>			
	030670	000167	176504			JMP	\$5055
5955	030674			ENDIF			
	030674					\$51244:	
5956							
5957							

;\*

CVTNP

```

5958 ; DO ONE INLINE CASE - CVTNPI
5959 ;-
5960
5961 030674 CALL CLRREG
      030674 004767 025540 JSR PC,CLRREG
5962 030700 CALL CLRDSB
      030700 004767 025770 JSR PC,CLRDSB
5963 030704 INLINE <CVTNPI>
      030704 076155 CVTNPI
5964 030706 INLINE <.WORD TCVNPI>
      030706 031266 .WORD TCVNPI
5965 030710 INLINE <.WORD TCVNPI+4>
      030710 031272 .WORD TCVNPI+4
5966 030712 LET CC := PSW
      030712 016767 147060 147600 MOV PSW,CC
5967 030720 LET CC := CC CLR.BY #177760
      030720 042767 177760 147572 BIC #177760,CC
5968 030726 IF CC NE #0 THEN
      030726 005767 147566 TST CC
      030732 001402 BEQ $51245
5969 030734 CALL ERROR
      030734 004767 026740 JSR PC,ERROR
5970 030740 ENDIF
5971 030740 IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN
      030740 005700 TST R0
      030742 001004 BNE $51246
      030744 005701 TST R1
      030746 001002 BNE $51246
      030750 005702 TST R2
      030752 001402 BEQ $51247
      030754 $51246:
5972 030754 CALL ERROR
      030754 004767 026720 JSR PC,ERROR
5973 030760 ENDIF
5974 030760 IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
      030760 005703 TST R3
      030762 001004 BNE $51250
      030764 005704 TST R4
      030766 001002 BNE $51250
      030770 005705 TST R5
      030772 001402 BEQ $51251
      030774 $51250:
5975 030774 CALL ERROR
      030774 004767 026700 JSR PC,ERROR
5976 031000 ENDIF
5977 031000 IFB DS.DST NE #124 THEN
      031000 126727 033540 000124 CMPB DS.DST,#124
      031006 001402 BEQ $51252
5978 031010 CALL ERROR
      031010 004767 026664 JSR PC,ERROR
5979 031014 ENDIF
5980 031014 IFB DS.DST+1 NE #62 THEN
      031014 126727 033525 000062 CMPB DS.DST+1,#62

```



CVTNP

```

5981 031022 001402                                BEQ    $51253
      031024                                CALL ERROR
5982 031024 004767 026650                        JSR    PC.ERROR
      031030                                ENDIF
5983 031030                                $51253:
      031030 126727 033512 000037                IFB DS.DST+2 NE #37 THEN
      031036 001402                                CMPB   DS.DST+2,#37
5984 031040                                BEQ    $51254
      031040                                CALL ERROR
5985 031040                                JSR    PC.ERROR
      031044                                ENDIF
5986 031044                                $51254:
      031044 000167 001552                        INLINE <JMP CISSEND>
                                           JMP   CISSEND

```

```

5987
5988 ;*
5989 ; TABLE OF TEST CASES FOR CVTNP
5990 ;
5991

```

5992	031050	000010	TCVTNP: .WORD	8.		;# OF TEST CASES FOR CVTNP
5993						
5994	031052	000000		WORD	0	;SRC DATA TYPE AND LENGTH
5995	031054	031140		.WORD	CVTNP1	;POINTER TO INTERMEDIATE SRC STRING
5996	031056	060000		.WORD	60000	;DST DATA TYPE AND LENGTH
5997						
5998	031060	050000		.WORD	50000	;TEST CASE NUMBER 2
5999	031062	031144		.WORD	CVTNP2	
6000	031064	060000		.WORD	60000	
6001						
6002	031066	030000		.WORD	30000	;TEST CASE NUMBER 3
6003	031070	031150		.WORD	CVTNP3	
6004	031072	060037		.WORD	60037	
6005						
6006	031074	020023		.WORD	20023	;TEST CASE NUMBER 4
6007	031076	031154		.WORD	CVTNP4	
6008	031100	060000		.WORD	60000	
6009						
6010	031102	040010		.WORD	40010	;TEST CASE NUMBER 5
6011	031104	031202		.WORD	CVTNP5	
6012	031106	060020		.WORD	60020	
6013						
6014	031110	010011		.WORD	10011	;TEST CASE NUMBER 6
6015	031112	031214		.WORD	CVTNP6	
6016	031114	060006		.WORD	60006	
6017						
6018	031116	000017		.WORD	17	;TEST CASE NUMBER 7
6019	031120	031230		.WORD	CVTNP7	
6020	031122	060017		.WORD	60017	
6021						
6022	031124	000006		.WORD	6	;TEST CASE NUMBER 8
6023	031126	051252		.WORD	CVTNP8	
6024	031130	070006		.WORD	70006	
6025						
6026	031132	050000		.WORD	50000	;TEST CASE NUMBER 9
6027	031134	031262		.WORD	CVTNP9	
6028	031136	060002		.WORD	60002	
6029						

CVTNP

6030											
6031	031140	000000									
6032	031142	000									
6033											
6034											
6035	031144	177777									
6036	031146	000									
6037											
6038											
6039	031150	177777									
6040	031152	000									
6041											
6042											
6043	031154	177777									
6044	031156	011	010	007							
	031161	006	005	004							
	031164	003	002	001							
	031167	000	001	002							
	031172	003	004	005							
	031175	006									
6045	031176	007	010	011							
6046											
6047											
6048	031202	000000									
6049	031204	007	000	006							
	031207	001	005	002							
	031212	004	003								
6050											
6051											
6052	031214	177777									
6053	031216	001	003	005							
	031221	007	011	002							
	031224	004	006	010							
6054											
6055											
6056	031230	177777									
6057	031232	005	005	007							
	031235	003	010	002							
	031240	001	011	004							
	031243	007	006	000							
	031246	000	007	001							
6058											
6059											
6060	031252	177777									
6061	031254	007	005	003							
	031257	001	004	002							
6062											
6063											
6064	031262	000000									
6065	031264	007	002								
6066											
6067											
6068											
6069	031266										
6070	031266	000005									
6071	031270	031276									
6072	031272	070005									

```

CVTNP1: .WORD 0 ;SIGN WORD: 0 = ., 1 =
        .BYTE ;MAGNITUDE, ONE BCD PER BYTE
        .EVEN

CVTNP2: .WORD -1 ;INTERMEDIATE SRC STRING 2
        .BYTE
        .EVEN

CVTNP3: .WORD -1 ;INTERMEDIATE SRC STRING 3
        .BYTE
        .EVEN

CVTNP4: .WORD -1 ;INTERMEDIATE SRC STRING 4
        .BYTE 011,010,007,006,005,004,003,002,001,000,001,002,003,004,005,006

        .BYTE 007,010,011
        .EVEN

CVTNP5: .WORD 0 ;INTERMEDIATE SRC STRING 5
        .BYTE 007,000,006,001,005,002,004,003
        .EVEN

CVTNP6: .WORD -1 ;INTERMEDIATE SRC STRING 6
        .BYTE 001,003,005,007,011,002,004,006,010
        .EVEN

CVTNP7: .WORD -1 ;INTERMEDIATE SRC STRING 7
        .BYTE 005,005,007,003,010,002,001,011,004,007,006,000,000,007,001
        .EVEN

CVTNP8: .WORD -1 ;INTERMEDIATE SRC STRING 8
        .BYTE 007,005,003,001,004,002
        .EVEN

CVTNP9: .WORD 0 ;INTERMEDIATE SRC STRING 9
        .BYTE 007,002
        .EVEN

TCVNPI: ;INLINE TEST CASE
        .WORD 000005 ;SIGNED ZONED, LENGTH = 5
        .WORD NP1 ;POINTER TO SRC
        .WORD 070005 ;UNSIGNED PACKED, LENGTH = 5

```



CVTNP

```

6079 ;*****
6080 .SBTTL CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE
6081 ;*****
6082 ;* THIS ROUTINE CONVERTS A LONG INTEGER INTO
6083 ;* AN INTERMEDIATE FORM.
6084 ;* THE INPUT IS THE 31 BIT LONG INTEGER (PLUS THE
6085 ;* SIGN BIT) IN REGISTERS R2 AND R3:
6086 ;*
6087 ;*
6088 ;*      R2  ! SIGN ! ----- HIGH ----- !
6089 ;*      -----
6090 ;*
6091 ;*
6092 ;*      R3  ! ----- LOW ----- !
6093 ;*      -----
6094 ;*
6095 ;*
6096 ;* THE OUTPUT IS:
6097 ;* SIGNO SET TO SIGN OF THE LONG INTEGER
6098 ;* (0 FOR +, 177777 FOR -); AND 10 BINARY
6099 ;* CODED DECIMAL (BCD) DIGITS PUT INTO MEMORY.
6100 ;* ONE DIGIT PER BYTE, STARTING AT DSBUF1.
6101 ;* MOST SIGNIFICANT DIGIT FIRST. DSLEN1 WILL
6102 ;* HAVE THE LENGTH (EXCLUDING LEADING ZEROES).
6103 ;* THE GENERAL PURPOSE REGISTERS ARE SAVED
6104 ;* AT THE BEGINNING OF THE ROUTINE AND RESTORED
6105 ;* AT THE END.
6106 ;* DSBUF1 DECIMAL STRING BUFFER SHOULD BE CLEAR BEFORE
6107 ;* ENTERING THIS ROUTINE.
6108
6109 031304 ROUTINE CVTLI
        031304
6110
6111 ;*
6112 ; ROUTINE INITIALIZATION
6113 ;-
6114
6115 031304 CALL SAVREG ;SAVE THE GPR'S
        031304 004767 025426 JSR PC.SAVREG
6116 031310 LET SIGNO := #0 CLR SIGNO
        031310 005067 147134
6117 031314 LET DSLEN1 := #0 CLR DSLEN1
        031314 005067 147174
6118
6119 ;*
6120 ; TEST FOR SPECIAL CASES OF 0 VALUE OR MOST NEGATIVE VALUE
6121 ;-
6122
6123 031320 INLINE <TST R2>
        031320 005702 TST R2
6124 031322 IFCOND MI THEN
        031322 100003 BPL $51257
6125 031324 LET SIGNO := #-1 MOV #-1,SIGNO
        031324 012767 177777 14116
6126 031332 ENDIF
6127 031332 LET TEMPO := R2 CLR.BY #100000 $51257:

```

CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE

```

031332 010267 147140
031336 042767 100000 147132
6128 031344 IF R3 EQ #0 AND TEMPO EQ #0 THEN
031344 005703
031346 001044
031350 005767 147122
031354 001041
6129 031356 IF SIGN0 EQ #0 THEN
031356 005767 147066
031362 001003
6130 ;RETURN
6131 031364 INLINE <JMP XCVTLI>
031364 000167 000616
6132 031370 ELSE ;RESULT - -2,147,483,648
031370 000433 ;51261:
031372
6133 031372 LET R0 := #DSBUF1
031372 012700 064440
6134 031376 LET (R0)+ :B= #2
031376 112720 000002
6135 031402 LET (R0)+ :B= #1
031402 112720 000001
6136 031406 LET (R0)+ :B= #4
031406 112720 000004
6137 031412 LET (R0)+ :B= #7
031412 112720 000007
6138 031416 LET (R0)+ :B= #4
031416 112720 000004
6139 031422 LET (R0)+ :B= #8.
031422 112720 000010
6140 031426 LET (R0)+ :B= #3
031426 112720 000003
6141 031432 LET (R0)+ :B= #6
031432 112720 000006
6142 031436 LET (R0)+ :B= #4
031436 112720 000004
6143 031442 LET (R0)+ :B= #8.
031442 112720 000010
6144 031446 LET DSLEN1 := #10.
031446 012767 000012 147040
6145 ;RETURN
6146 031454 INLINE <JMP XCVTLI>
031454 000167 000526
6147 031460 ENDIF
031460 ;51262:
6148 031460 ENDIF ;51260:
031460
6149
6150 ;+
6151 ; LONG INTEGER SOURCE WAS NOT ZERO OR MOST NEGATIVE NUMBER.
6152 ; SO PREPARE FOR CONVERSION ALGORITHM.
6153 ;-
6154
6155 031460 IF SIGN0 EQ #-1 THEN
031460 026727 146764 177777
031466 001005
6156 031470 LET R3 := NEGATE R3

```

```

MOV R2,TEMPO
BIC #100000,TEMPO
TST R3
BNE $51260
TST TEMPO
BNE $51260
TST SIGN0
BNE $51261
JMP XCVTLI
BR $51262
MOV #DSBUF1,R0
MOVB #2,(R0)+
MOVB #1,(R0)+
MOVB #4,(R0)+
MOVB #7,(R0)+
MOVB #4,(R0)+
MOVB #8.,(R0)+
MOVB #3,(R0)+
MOVB #6,(R0)+
MOVB #4,(R0)+
MOVB #8.,(R0)+
MOV #10.,DSLEN1
JMP XCVTLI

```

## CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE

6157	031470	005403				NEG	R3
	031472			LET R2 := COMP R2		COM	R2
6158	031472	005102		IF R3 EQ #0 THEN		TST	R3
	031474	005703				BNE	\$51264
	031476	001001		LET R2 := R2 + #1		INC	R2
6159	031500						
	031500	005202		ENDIF			
6160	031502			ENDIF		\$51264:	
	031502					\$51263:	
6161	031502			LET R0 := #DSBUF1		MOV	#DSBUF1,R0
6162	031502	012700	064440				
6163							
6164				;			
6165				;	GET TENTH BCD DIGIT (MOST SIGNIFICANT DIGIT)		
6166				;			
6167				;			
6168	031506			WHILE R2 HI #35632 DO		\$51265:	
	031506					CMP	R2,#35632
	031506	020227	035632			BLOS	\$51266
	031512	101407				SUB	#145000,R3
6169	031514			LET R3 := R3 - #145000		SBC	R2
	031514	162703	145000	INLINE <SBC R2>		SUB	#35632,R2
6170	031520			LET R2 := R2 - #35632		INCB	(R0)
	031520	005602		LET (R0) := (R0) + #1		BR	\$51265
6171	031522			ENDDO		\$51266:	
	031522	162702	035632	IF R2 EQ #35632 AND R3 HIS #145000 THEN		CMP	R2,#35632
6172	031526					BNE	\$51267
	031526	105210				CMP	R3,#145000
6173	031530			LET R3 := R3 - #145000		BLO	\$51267
	031530	000766		LET R2 := #0		SUB	#145000,R3
6174	031532			LET (R0) := (R0) + #1		CLR	R2
	031532			ENDIF		INCB	(R0)
	031532	020227	035632			\$51267:	
	031536	001007					
	031540	020327	145000				
	031544	103404					
6175	031546			LET R3 := R3 - #145000			
	031546	162703	145000	LET R2 := #0			
6176	031552			LET (R0) := (R0) + #1			
	031552	005002					
6177	031554			ENDIF			
	031554	105210					
6178	031556						
	031556						
6179				;			
6180				;	GET NINTH BCD DIGIT		
6181				;			
6182				;			
6183				;			
6184	031556			LET R0 := R0 + #1		INC	R0
	031556	005200		WHILE R2 HI #2765 DO			
6185	031560					\$51270:	
	031560					CMP	R2,#2765
	031560	020227	002765				

CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE

```

6186 031564 101407          LET R3 := R3 - #160400          BLOS  $51271
6187 031566 162703 160400  INLINE <SBC R2>          SUB   #160400,R3
6188 031572 005602          SBC R2
6189 031574 162702 002765  LET R2 := R2 - #2765          SUB   #2765,R2
6190 031600 105210          LET (R0) :B= (R0) + #1       INCB  (R0)
6191 031602 000766          ENDDO
6191 031604          IF R2 EQ #2765 AND R3 HIS #160400 THEN $51271:
6191 031604 020227 002765          CMP   R2,#2765
6191 031610 001007          BNE  $51272
6191 031612 020327 160400          CMP   R3,#160400
6191 031616 103404          BLO  $51272
6192 031620          LET R3 := R3 - #160400          SUB   #160400,R3
6193 031624 162703 160400  LET R2 := #0          CLR   R2
6194 031626 005002          LET (R0) :B= (R0) + #1       INCB  (R0)
6195 031630 105210          ENDF
6196 031630          $51272:
6197          ;+
6198          ; GET EIGHTH BCD DIGIT
6199          ;-
6201 031630          LET R0 := R0 + #1
6202 031632 005200          WHILE R2 HI #230 DO          INC   R0
6202 031632 020227 000230          $51273:
6202 031636 101407          CMP   R2,#230
6203 031640          LET R3 := R3 - #113200          BLOS  $51274
6204 031644 162703 113200  INLINE <SBC R2>          SUB   #113200,R3
6205 031646 005602          SBC R2
6206 031646 162702 000230  LET R2 := R2 - #230          SUB   #230,R2
6207 031652 105210          LET (R0) :B= (R0) + #1       INCB  (R0)
6208 031654 000766          ENDDO
6208 031656          IF R2 EQ #230 AND R3 HIS #113200 THEN $51274:
6208 031656 020227 000230          CMP   R2,#230
6208 031662 001007          BNE  $51275
6208 031664 020327 113200          CMP   R3,#113200
6208 031670 103404          BLO  $51275
6209 031672          LET R3 := R3 - #113200          SUB   #113200,R3
6210 031676 162703 113200  LET R2 := #0          CLR   R2
6211 031700 005002          LET (R0) :B= (R0) + #1

```

CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE

6212	031700 105210	ENDIF	INCB (R0)
	031702		\$51275:
6213	031702		
6214		;* GET SEVENTH BCD DIGIT	
6215		;	
6216		;	
6217			
6218	031702	LET R0 := R0 + #1	
	031702 005200		INC R0
6219	031704	WHILE R2 HI #17 DO	\$51276:
	031704		CMP R2,#17
	031704 020227 000017		BLOS \$51277
	031710 101407		SUB #41100,R3
6220	031712	LET R3 := R3 - #41100	SBC R2
	031712 162703 041100		SUB #17,R2
6221	031716	INLINE <SBC R2>	INCB (R0)
	031716 005602		BR \$51276
6222	031720	LET R2 := R2 - #17	\$51277:
	031720 162702 000017		CMP R2,#17
6223	031724	LET (R0) :B= (R0) + #1	BNE \$51300
	031724 105210		CMP R3,#41100
6224	031726	ENDDO	BLO \$51300
	031726 000766		SUB #41100,R3
	031730		CLR R2
6225	031730	IF R2 EQ #17 AND R3 HIS #41100 THEN	INCB (R0)
	031730 020227 000017		\$51300:
	031734 001007		
	031736 020327 041100		
	031742 103404		
6226	031744	LET R3 := R3 - #41100	
	031744 162703 041100		
6227	031750	LET R2 := #0	
	031750 005002		
6228	031752	LET (R0) :B= (R0) + #1	
	031752 105210		
6229	031754	ENDIF	
	031754		
6230			
6231		;* GET SIXTH BCD DIGIT	
6232		;	
6233		;	
6234		;-	
6235	031754	LET R0 := R0 + #1	
	031754 005200		INC R0
6236	031756	WHILE R2 HI #1 DO	\$51301:
	031756		CMP R2,#1
	031756 020227 000001		BLOS \$51302
	031762 101406		SUB #103240,R3
6237	031764	LET R3 := R3 - #103240	SBC R2
	031764 162703 103240		DEC R2
6238	031770	INLINE <SBC R2>	INCB (R0)
	031770 005602		
6239	031772	LET R2 := R2 - #1	
	031772 005302		
6240	031774	LET (R0) :B= (R0) + #1	
	031774 105210		



CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE

6241	031776		ENDDO		
	031776	000767			
	032000				
6242	032000		IF R2 EQ #1 AND R3 HIS #103240 THEN	\$51302:	BR \$51301
	032000	020227	000001		
	032004	001007			CMP R2,#1
	032006	020327	103240		BNE \$51303
	032012	103404			CMP R3,#103240
6243	032014		LET R3 := R3 - #103240		BLO \$51303
	032014	162703	103240		
6244	032020		LET R2 := #0		SUB #103240,R3
	032020	005002			CLR R2
6245	032022		LET (R0) :B= (R0) + #1		INCB (R0)
	032022	105210			
6246	032024		ENDIF	\$51303:	
	032024				
6247					
6248					
6249			; ; GET FIFTH BCD DIGIT		
6250			; ;-		
6251					
6252	032024		LET R0 := R0 + #1		
	032024	005200			INC R0
6253	032026		WHILE R2 EQ #1 DO	\$51304:	
	032026				
	032026	020227	000001		CMP R2,#1
	032032	001005			BNE \$51305
6254	032034		LET R3 := R3 - #23420		SUB #23420,R3
	032034	162703	023420		
6255	032040		INLINE <SBC R2>		SBC R2
	032040	005602			
6256	032042		LET (R0) :B= (R0) + #1		INCB (R0)
	032042	105210			
6257	032044		ENDDO		BR \$51304
	032044	000770		\$51305:	\$51304
	032046				
6258	032046		WHILE R3 HIS #23420 DO	\$51306:	
	032046				
	032046	020327	023420		CMP R3,#23420
	032052	103404			BLO \$51307
6259	032054		LET R3 := R3 - #23420		SUB #23420,R3
	032054	162703	023420		
6260	032060		LET (R0) :B= (R0) + #1		INCB (R0)
	032060	105210			
6261	032062		ENDDO		BR \$51306
	032062	000771		\$51307:	\$51306
	032064				
6262					
6263			; ; GET FOURTH BCD DIGIT		
6264			; ;-		
6265					
6266					
6267	032064		LET R0 := R0 + #1		INC R0
	032064	005200			
6268	032066		WHILE R3 HIS #1750 DO	\$51310:	
	032066				
	032066	020327	001750		CMP R3,#1750

CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE

6269	032072	103404							
	032074			LET R3 := R3 - #1750		BLO	\$51311		
6270	032074	162703	001750			SUB	#1750,R3		
	032100			LET (R0) :B= (R0) + #1		INCB	(R0)		
6271	032100	105210				BR	\$51310		
	032102			ENDDO		\$51311:			
	032102	000771							
	032104								
6272									
6273				;* GET THIRD BCD DIGIT					
6274				;					
6275				;-					
6276									
6277	032104			LET R0 := R0 + #1					
	032104	005200				INC	R0		
6278	032106			WHILE R3 HIS #144 DO		\$51312:			
	032106								
	032106	020327	000144			CMP	R3,#144		
	032112	103404				BLO	\$51313		
6279	032114			LET R3 := R3 - #144					
	032114	162703	000144			SUB	#144,R3		
6280	032120			LET (R0) :B= (R0) + #1		INCB	(R0)		
	032120	105210							
6281	032122			ENDDO		BR	\$51312		
	032122	000771				\$51313:			
	032124								
6282									
6283				;* GET SECOND AND FIRST BCD DIGITS					
6284				;					
6285				;-					
6286									
6287	032124			LET R0 := R0 + #1					
	032124	005200				INC	R0		
6288	032126			WHILE R3 HIS #12 DO		\$51314:			
	032126								
	032126	020327	000012			CMP	R3,#12		
	032132	103404				BLO	\$51315		
6289	032134			LET R3 := R3 - #12					
	032134	162703	000012			SUB	#12,R3		
6290	032140			LET (R0) :B= (R0) + #1		INCB	(R0)		
	032140	105210							
6291	032142			ENDDO		BR	\$51314		
	032142	000771				\$51315:			
	032144								
6292	032144			LET R0 := R0 + #1					
	032144	005200				INC	R0		
6293	032146			LET (R0) :B= R3		MOVB	R3,(R0)		
	032146	110310							
6294									
6295				;* DETERMINE THE NUMBER OF SIGNIFICANT DIGITS.					
6296				;					
6297				;					
6298									
6299	032150			LET R0 := #DSBUF1					
	032150	012700	064440			MOV	#DSBUF1,R0		
6300	032154			WHILEB (R0) EQ #0 DO					
	032154					\$51316:			

CONVERT LONG INTEGER TO INTERMEDIATE FORM ROUTINE

```

032154 105710
032156 001002
6301 032160          LET RO := RO + #1
032160 005200
6302 032162          ENDDO
032162 000774
032164
6303 032164          LET RO := RO - #DSBUF1
032164 162700 064440          ;# OF LEADING 0'S
6304 032170          LET DSLEN1 := #10. - RO
032170 012767 000012 146316          ;# OF MOST SIGNIFICANT DIGITS
032176 160067 146312          MOV #10.,DSLEN1
6305 032202          CALL RESREG          ;RESTORE THE GPR'S
032202 004767 024562          SUB RO,DSLEN1
6306 032206          INLINE <XCVTLI:>
032206
6307 032206          ENDRTN
032206          XCVTLI:
032206          $51255:
032206 000207          $51256:
6308          RTS PC
6309
6310          ;:*****
6311          .SBTTL CONVERT INTERMEDIATE FORM TO LONG INTEGER ROUTINE
6312          ;:*****
6313          ;* THIS ROUTINE CONVERTS A 10 BCD DIGIT INTERMEDIATE FORM INTO
6314          ;* LONG INTEGER FORMAT. THE INPUT IS 10 BCD DIGITS IN THE 10 BYTES STARTING
6315          ;* AT DSBUF1, MOST SIGNIFICANT DIGIT FIRST; AND SIGNO.
6316          ;* THE OUTPUT IS A TWO'S COMPLEMENT 31 BIT BINARY LONG INTEGER IN LOCATIONS
6317          ;* R2.EXP AND R3.EXP.
6318          ;* THIS ROUTINE ASSUMES THAT THE 10 BCD DIGIT NUMBER IS SMALL ENOUGH
6319          ;* TO BE REPRESENTED AS A 31 BIT BINARY INTEGER.
6320 032210          ROUTINE CVTIL
032210          CVTIL:
6321
6322          ;+
6323          ; ROUTINE INITIALIZATION
6324          ;-
6325
6326 032210          CALL SAVREG          ;SAVE THE GPR'S
032210 004767 024522          JSR PC,SAVREG
6327 032214          LET R2.EXP := #0
032214 005067 146310          CLR R2.EXP
6328 032220          LET R3.EXP := #0
032220 005067 146306          CLR R3.EXP
6329 032224          LET RO := #DSBUF1
032224 012700 064440          MOV #DSBUF1,RO
6330
6331          ;+
6332          ;+
6333          ; ADD IN BINARY EQUIVALENT OF TENTH BCD DIGIT
6334          ;-
6335
6336 032230          WHILEB (RO) GT #0 DO
032230
032230 105710          $51322:
032232 003412          TSTB (RO)
          BLE $51323

```

## CONVERT INTERMEDIATE FORM TO LONG INTEGER ROUTINE

```

6337 032234          LET R3.EXP := R3.EXP + #145000
      032234 062767 145000 146270
6338 032242          ADD      #145000,R3.EXP
      032242 005567 146262          ADC  R2.EXP
6339 032246          LET R2.EXP := R2.EXP + #35632
      032246 062767 035632 146254
6340 032254          ADD      #35632,R2.EXP
      032254 105310          DECB  (R0)
6341 032256          LET (R0) :B= (R0) - #1
      032256 000764          ENDDO
      032260          BR      #51322
6342          #51323:
6343          ;*
6344          ; ADD IN BINARY EQUIVALENT OF NINTH BCD DIGIT
6345          ;-
6346
6347 032260          LET R0 := R0 + #1
      032260 005200          WHILEB (R0) GT #0 DO
6348 032262          INC      R0
      032262          #51324:
      032262 105710          TSTB  (R0)
      032264 003412          BLE  #51325
6349 032266          LET R3.EXP := R3.EXP + #160400
      032266 062767 160400 146236
6350 032274          ADD      #160400,R3.EXP
      032274 005567 146230          ADC  R2.EXP
6351 032300          LET R2.EXP := R2.EXP + #2765
      032300 062767 002765 146222
6352 032306          ADD      #2765,R2.EXP
      032306 105310          DECB  (R0)
6353 032310          ENDDO
      032310 000764          BR      #51324
      032312          #51325:
6354
6355          ;*
6356          ; ADD IN BINARY EQUIVALENT OF EIGHTH BCD DIGIT
6357          ;-
6358
6359 032312          LET R0 := R0 + #1
      032312 005200          WHILEB (R0) GT #0 DO
6360 032314          INC      R0
      032314          #51326:
      032314 105710          TSTB  (R0)
      032316 003412          BLE  #51327
6361 032320          LET R3.EXP := R3.EXP + #113200
      032320 062767 113200 146204
6362 032326          ADD      #113200,R3.EXP
      032326 005567 146176          ADC  R2.EXP
6363 032332          LET R2.EXP := R2.EXP + #230
      032332 062767 000230 146170
6364 032340          ADD      #230,R2.EXP
      032340 105310          DECB  (R0)
6365 032342          ENDDO
      032342 000764          BR      #51326
      032344          #51327:
6366
6367          ;*

```

CONVERT INTERMEDIATE FORM TO LONG INTEGER ROUTINE

```

6368 ; ADD IN BINARY EQUIVALENT OF SEVENTH BCD DIGIT
6369 ; -
6370
6371 032344 LET R0 := R0 + #1
032344 005200
6372 032346 WHILEB (R0) GT #0 DO
032346
032346 105710 #51330:
032350 003412 TSTB (R0)
6373 032352 LET R3.EXP := R3.EXP + #41100 BLE #51331
032352 062767 041100 146152 ADD #41100,R3.EXP
6374 032360 INLINE <ADC R2.EXP> ADC R2.EXP
032360 005567 146144 LET R2.EXP := R2.EXP + #17 ADD #17,R2.EXP
6375 032364 LET R2.EXP := R2.EXP + #17 ADD #17,R2.EXP
032364 062767 000017 146136 DEC B (R0)
6376 032372 LET (R0) :B= (R0) - #1 BR #51330
032372 105310 ENDDO #51331:
6377 032374
032374 000764
032376
6378
6379 ; *
6380 ; ADD IN BINARY EQUIVALENT OF SIXTH BCD DIGIT
6381 ; -
6382
6383 032376 LET R0 := R0 + #1
032376 005200 INC R0
6384 032400 WHILEB (R0) GT #0 DO
032400
032400 105710 #51332:
032402 003411 TSTB (R0)
6385 032404 LET R3.EXP := R3.EXP + #103240 BLE #51333
032404 062767 103240 146120 ADD #103240,R3.EXP
6386 032412 INLINE <ADC R2.EXP> ADC R2.EXP
032412 005567 146112 LET R2.EXP := R2.EXP + #1 INC R2.EXP
6387 032416 LET R2.EXP := R2.EXP + #1 INC R2.EXP
032416 005267 146106 LET (R0) :B= (R0) - #1 DEC B (R0)
6388 032422 LET (R0) :B= (R0) - #1 BR #51332
032422 105310 ENDDO #51333:
6389 032424
032424 000765
032426
6390
6391 ; *
6392 ; ADD IN BINARY EQUIVALENT OF FIFTH BCD DIGIT
6393 ; -
6394
6395 032426 LET R0 := R0 + #1
032426 005200 INC R0
6396 032430 WHILEB (R0) GT #0 DO
032430
032430 105710 #51334:
032432 003407 TSTB (R0)
6397 032434 LET R3.EXP := R3.EXP + #23420 BLE #51335
032434 062767 023420 146070 ADD #23420,R3.EXP
6398 032442 INLINE <ADC R2.EXP> ADC R2.EXP
032442 005567 146062

```

CONVERT INTERMEDIATE FORM TO LONG INTEGER ROUTINE

6399	032446			LET (R0) :B= (R0) - #1		
	032446	105310				DECB (R0)
6400	032450			ENDDO		BR \$51334
	032450	000767				\$51335:
	032452					
6401						
6402						
6403				;* ; ADD IN BINARY EQUIVALENT OF FOURTH BCD DIGIT		
6404				;-		
6405						
6406	032452			LET R0 := R0 + #1		
	032452	005200				INC R0
6407	032454			WHILEB (R0) GT #0 DO		\$51336:
	032454					TSTB (R0)
	032454	105710				BLE \$51337
	032456	003407				
6408	032460			LET R3.EXP := R3.EXP + #1750		
	032460	062767	001750	146044		ADD #1750,R3.EXP
6409	032466			INLINE <ADC R2.EXP>		
	032466	005567	146036			ADC R2.EXP
6410	032472			LET (R0) :B= (R0) - #1		
	032472	105310				DECB (R0)
6411	032474			ENDDO		BR \$51336
	032474	000767				\$51337:
	032476					
6412						
6413						
6414				;* ; ADD IN BINARY EQUIVALENT OF THIRD BCD DIGIT		
6415				;-		
6416						
6417	032476			LET R0 := R0 + #1		
	032476	005200				INC R0
6418	032500			WHILEB (R0) GT #0 DO		\$51340:
	032500					TSTB (R0)
	032500	105710				BLE \$51341
	032502	003407				
6419	032504			LET R3.EXP := R3.EXP + #144		
	032504	062767	000144	146020		ADD #144,R3.EXP
6420	032512			INLINE <ADC R2.EXP>		
	032512	005567	146012			ADC R2.EXP
6421	032516			LET (R0) :B= (R0) - #1		
	032516	105310				DECB (R0)
6422	032520			ENDDO		BR \$51340
	032520	000767				\$51341:
	032522					
6423						
6424						
6425				;* ; ADD IN BINARY EQUIVALENT OF SECOND BCD DIGIT		
6426				;-		
6427						
6428	032522			LET R0 := R0 + #1		
	032522	005200				INC R0
6429	032524			WHILEB (R0) GT #0 DO		\$51342:
	032524					TSTB (R0)
	032524	105710				BLE \$51343
	032526	003407				
6430	032530			LET R3.EXP := R3.EXP + #12		



CONVERT INTERMEDIATE FORM TO LONG INTEGER ROUTINE

```

6456 ;*****
6457 ;*****
6458 ;**
6459 ;           C I S 6   M O D U L E
6460 ;
6461 ;   THIS MODULE TESTS THE NUMERIC DATA TYPE ARITHMETIC
6462 ;   INSTRUCTIONS: ADDN, SUBN, CMPN, AND ASHN.
6463 ;
6464 ;*****
6465 ;*****
6466 ;
6467 ;
6468 ;
6469 ;.SBTTL  CIS6 INITIALIZATION
6470 ;*****
6471 ;
6472 ;   CIS6 INITIALIZATION
6473 ;
6474 ;*****
6475 ;
6476 ;IF SWR<5:0> ARE CLEARED OR IF SWR<2> IS SET THEN EXECUTE
6477 ; THE CIS6 MODULE; OTHERWISE, SKIP TO THE NEXT CIS MODULE.
6478 ;
6479 032622      INLINE  <SCOPE>
6480 032622 000004      SCOPE
6481 032624      INLINE <CKSWR>
6482 032624 104406      CKSWR
6483 032626      LET TEMPO := @SWR CLR.BY #177700
6484 032626 017767 145574 145642      MOV @SWR,TEMPO
6485 032634 042767 177700 145634      BIC #177700,TEMPO
6486 032642      IF TEMPO NE #0 THEN
6487 032642 005767 145630      TST TEMPO
6488 032646 001406      BEQ $51346
6489 032650      IF #BIT2 NOTSETIN TEMPO THEN
6490 032650 032767 000004 145620      BIT #BIT2,TEMPO
6491 032656 001002      BNE $51347
6492 032660      INLINE  <JMP  CIS6END>
6493 032660 000167 012502      JMP  CIS6END
6494 032664      ENDIF
6495 032664      ENDIF $51347:
6496 032664      LET CTL := #6 ;CONTROL CHIP NUMBER $51346:
6497 032664 012767 000006 145654      MOV #6,CTL
6498 032672      LET ERRFLG := #0 ;INITIALIZE ERROR FLAG
6499 032672 005067 145560      CLR ERRFLG
6500 032676      LET PSW := #340
6501 032676 012767 000340 145072      MOV #340,PSW
6502 032704      LET ERR := #1
6503 032704 012767 000001 145640      MOV #1,ERR
6491
6492

```



ADDN

```

6494      .SBTTL  ADDN
6495      ;*****
6496      ;
6497      ;       ADDN TEST
6498      ;
6499      ;*****
6500
6501 032712 ROUTINE XADDN
6501 032712
6502 032712          XADDN:
6502 032712 000004          INLINE <SCOPE>          SCOPE
6503 032714          INLINE <NOP>          NOP
6504 032716          LET $TIMES := #10          ;ITERATION COUNT
6504 032716 012767 000010 145460          MOV      #10,$TIMES
6505 032724          LET COUNT := TADD          ;# OF TEST CASES
6505 032724 016767 032320 145536          MOV      TADD,COUNT
6506 032732          LET TSTCAS := #TADD*2          ;POINT TO FIRST TEST CASE
6506 032732 012767 065252 145604          MOV      #TADD*2,TSTCAS
6507
6508      ;*
6509      ;       CLR R0.EXP TO R3.EXP
6510      ;-
6511
6512 032740          INCR  R0 FROM #0 TO #6 BY #2
6512 032740 005000
6512 032742 000402          CLR      R0
6512 032744          $51353: BR      $51352
6512 032744 062700 000002          $51353: ADD      #2,R0
6512 032750          $51352: CMP      R0,#6
6512 032754 003003          $51352: BGT      $51354
6513 032756          INLINE <CLR R0.EXP(R0)>          CLR R0.EXP(R0)
6514 032762          ENDINC          BR      $51353
6514 032762 000770          $51354:
6515      ;REPEAT OUTER LOOP FOR TEST CASES
6516 032764          INLINE <$5050:>          $5050:
6517 032764          CALL CLRDSB          ;CLR DS BUFFERS          JSR      PC,CLRDSB
6517 032764 004767 023704
6518
6519      ;*
6520      ;       LOAD DATA0 AND CC.EXP FROM TEST CASE
6521      ;-
6522
6523 032770          LET R0 := TSTCAS
6523 032770 016700 145550          MOV      TSTCAS,R0
6524 032774          LET DATA0 := (R0)+          ;SRC1 LENGTH          MOV      (R0)+,DATA00
6524 032774 012067 145562          ;POINTER TO SRC1 DATA          MOV      (R0)+,DATA01
6525 033000          LET DATA01 := (R0)+          ;SRC2 LENGTH          MOV      (R0)+,DATA02
6526 033004          LET DATA02 := (R0)+          ;POINTER TO SRC2 DATA          MOV      (R0)+,DATA03
6527 033010          LET DATA03 := (R0)+          MOV      (R0)+,DATA03
6527 033010 012067 145554

```

```

ADDN
6528 033014          LET DATA04 := (R0)•          ;DEST LENGTH
      033014 012067 145552
6529 033020          LET DATA05 := (R0)•          ;POINTER TO DS.EXP DATA (SUM)
      033020 012067 145550
6530 033024          LET CC.EXP := (R0)•          ;CC.EXP (SUM)
      033024 012067 145472
6531
6532                ;+
6533                ;   LOAD DSBUF1
6534                ;-
6535
6536 033030          IF DATA00 NE #0 THEN          ;CHECK FOR ZERO LENGTH BUFFER
      033030 005767 145526
      033034 001413
6537 033036          LET R0 := DATA01
      033036 016700 145522
6538 033042          LET SIGN0 := (R0)•          ;STORE SIGN AND POINT AT FIRST BYTE
      033042 012067 145402
6539 033046          LET R1 := DATA00          ;# OF BYTES TO TRANSFER
      033046 016701 145510
6540 033052          LET R2 := #DSBUF1          ;TRANSFER DESTINATION
      033052 012702 064440
6541 033056          INLINE <1$:>
      033056
6542 033056          INLINE <MOVB (R0)•,(R2)•>
      033056 112022
6543 033060          INLINE <SOB R1,1$>
      033060 077102
6544 033062          ELSE
      033062 000402
      033064
6545 033064          LET SIGN0 := #0
      033064 005067 145360
6546 033070          ENDIF
      033070
6547
6548                ;+
6549                ;   LOAD DSBUF2
6550                ;-
6551
6552 033070          IF DATA02 NE #0 THEN
      033070 005767 145472
      033074 001413
6553 033076          LET R0 := DATA03
      033076 016700 145466
6554 033102          LET SIGN1 := (R0)•
      033102 012067 145344
6555 033106          LET R1 := DATA02
      033106 016701 145454
6556 033112          LET R2 := #DSBUF2
      033112 012702 064502
6557 033116          INLINE <2$:>
      033116
6558 033116          INLINE <MOVB (R0)•,(R2)•>
      033116 112022
6559 033120          INLINE <SOB R1,2$>
      033120 077102

TST DATA00
BEQ $51355
MOV DATA01,R0
MOV (R0)•,SIGN0
MOV DATA00,R1
MOV #DSBUF1,R2
1$:
MOVB (R0)•,(R2)•
SOB R1,1$
BR $51355:
CLR SIGN0
$51356:

TST DATA02
BEQ $51357
MOV DATA03,R0
MOV (R0)•,SIGN1
MOV DATA02,R1
MOV #DSBUF2,R2
2$:
MOVB (R0)•,(R2)•
SOB R1,2$

```

ADDN

```

6560 033122          ELSE
      033122 000402
      033124
6561 033124          LET SIGN1 := #0
      033124 005067 145322
6562 033130          ENDIF
      033130
6563
6564
6565
6566
6567
6568 033130          IF DATA04 NE #0 THEN
      033130 005767 145436
      033134 001415
6569 033136          LET R0 := DATA05
      033136 016700 145432
6570 033142          LET SIGN2 := (R0)+
      033142 012067 145306
6571 033146          LET R1 := DATA04
      033146 016701 145420
6572 033152          LET R2 := #DS.EXP
      033152 012702 064606
6573 033156          INLINE <3$:>
      033156
6574 033156          INLINE <MOVB (R0)+,(R2)>
      033156 112012
6575 033160          INLINE <BISB #60,(R2)+>
      033160 152722 000060
6576 033164          INLINE <SOB R1,3$>
      033164 077104
6577 033166          ELSE
      033166 000402
      033170
6578 033170          LET SIGN2 := #0
      033170 005067 145260
6579 033174          ENDIF
      033174
6580
6581
6582
6583
6584
6585 033174          LET DSTMP0 :B= DSBUF1
      033174 116767 031240 145352
6586 033202          LET TEMPO := #DSBUF1-1 + DATA00
      033202 012767 064437 145266
      033210 066767 145346 145260
6587 033216          LET DSTMP1 :B= @TEMPO
      033216 117767 145254 145331
6588 033224          LET DSTMP2 :B= DSBUF2
      033224 116767 031252 145324
6589 033232          LET TEMPO := #DSBUF2 1 + DATA02
      033232 012767 064501 145236
      033240 066767 145322 145230
6590 033246          LET DSTMP3 :B= @TEMPO
      033246 117767 145224 145303

```

```

BR $51360
$51357:
CLR SIGN1
$51360:
TST DATA04
BEQ $51361
MOV DATA05,R0
MOV (R0)+,SIGN2
MOV DATA04,R1
MOV #DS.EXP,R2
3$:
MOVB (R0)+,(R2)
BISB #60,(R2)+
SOB R1,3$
BR $51362
$51361:
CLR SIGN2
$51362:
MOVB DSBUF1,DSTMP0
MOV #DSBUF1-1,TEMPO
ADD DATA00,TEMPO
MOVB @TEMPO,DSTMP1
MOVB DSBUF2,DSTMP2
MOV #DSBUF2 1,TEMPO
ADD DATA02,TEMPO
MOVB @TEMPO,DSTMP3

```

ADDN

```

6591 033254          LET DSTMP4 :B= DS.EXP
      033254 116767 031326 145276
6592 033262          LET TEMPO := #DS.EXP 1 + DATA04
      033262 012767 064605 145206
      033270 066767 145276 145200
6593 033276          LET DSTMP5 :B= @TEMPO
      033276 117767 145174 145255
6594 033304          LET DESCR1 := #DSBUF1
      033304 012767 064440 145306
6595 033312          LET DESCR3 := #DSBUF2
      033312 012767 064502 145304
6596 033320          LET DESCR5 := #DS.DST
      033320 012767 064544 145302
6597
6598
6599 033326          ; INCR TEMPO FROM #1 TO #5 BY #1      SRC1 DATA TYPE LOOP
      033326 005037 000476          INLINE <CLR @TEMPO>
6600 033332          INLINE <INC1: INC TEMPO>
      033332 005267 145140
6601 033336          IF TEMPO GT #5 THEN
      033336 026727 145134 000005
      033344 003402
6602 033346          INLINE <JMP END1>
      033346 000167 001622
6603 033352          ENDIF
      033352
6604
6605
6606
6607
6608
6609 033352          LET DSBUF1 :B= DSTMP0          ;MSD
      033352 116767 145176 031060
6610 033360          LET TEMP2 := #DSBUF1-1 + DATA00
      033360 012767 064437 145114
      033366 066767 145170 145106
6611 033374          LET @TEMP2 :B= DSTMP1          ;LSD
      033374 116777 145155 145100
6612
6613
6614
6615
6616
6617 033402          SELECT TEMPO OF 5
      033402 016746 145070
      033406 006316
      033410 060716
      033412 063607
      033414
      033414 000012
      033416 000052
      033420 000116
      033422 000162
      033424 000226
6618 033426          CASE 1
      033426
6619 033426          LET DESCRO := DATA00          ;SIGNED ZONED

```

```

MOV B DS.EXP,DSTMP4
MOV #DS.EYP-1,TEMPO
ADD DATA04,TEMPO
MOV B @TEMPO,DSTMP5
MOV #DSBUF1,DESCR1
MOV #DSBUF2,DESCR3
MOV #DS.DST,DESCR5

```

```

CLR @TEMPO
INC1: INC TEMPO
CMP TEMPO,#5
BLE $51363
JMP END1

```

\$51363:

```

MOV B DSTMP0,DSBUF1
MOV #DSBUF1-1,TEMP2
ADD DATA00,TEMP2
MOV B DSTMP1,@TEMP2

```

\$51364:

```

MOV TEMPO,(SP)
ASL (SP)
ADD PC,(SP)
ADD @ (SP),PC
.WORD $51372 $51364
.WORD $51371 $51364
.WORD $51370 $51364
.WORD $51367 $51364
.WORD $51366 $51364

```

\$51372:

ADDN

```

6620 033426 016767 145130 145162          MOV      DATA00,DESCRO
      033434                                IF SIGNO EQ #0 THEN
      033434 005767 145010                                TST      SIGNO
      033440 001004                                BNE      $51373
6621 033442                                LET @TEMP2 :B= @TEMP2 SET.BY #60 ;+
      033442 152777 000060 145032                                BISS    #60,@TEMP2
6622 033450                                ELSE
      033450 000403                                BR       $51374
      033452                                $51373:
6623 033452                                LET @TEMP2 :B= @TEMP2 SET.BY #160 ;-
      033452 152777 000160 145022                                BISS    #160,@TEMP2
6624 033460                                ENDIF
      033460                                $51374:
6625 033460                                LET DSBUF1 1 :B= #0          ;FIX LEADING SEPARATE RESULT
      033460 105067 030753                                CLRB    DSBUF1 1
6626
6627 033464                                CASE 2
      033464 000512                                BR       $51365
      033466                                $51371:
6628 033466                                LET DESCRO := DATA00 SET.BY #20000 ;TRAILING OVERPUNCH
      033466 016767 145070 145122                                MOV      DATA00,DESCRO
      033474 052767 020000 145114                                BIS      #20000,DESCRO
6629 033502                                LET R5 :B= DSTMP1 ;GET LSD
      033502 116705 145047                                MOV      DSTMP1,R5
6630 033506                                IF SIGNO EQ #-1 THEN
      033506 026727 144736 177777                                CMP      SIGNO,#-1
      033514 001002                                BNE      $51375
6631 033516                                LET R5 := R5 + #10.          ;CHANGE INDEX FOR
      033516 062705 000012                                ADD      #10.,R5
6632 033522                                ENDIF
      033522                                $51375:
6633 033522                                LET @TEMP2 :B= OVPNCH(R5)
      033522 116577 056330 144752                                MOV      OVPNCH(R5) @TEMP2
6634
6635 033530                                CASE 3
      033530 000470                                BR       $51365
      033532                                $51370:
6636 033532                                LET DESCRO := DATA00 SET.BY #30000 ;LEADING OVERPUNCH
      033532 016767 145024 145056                                MOV      DATA00,DESCRO
      033540 052767 030000 145050                                BIS      #30000,DESCRO
6637 033546                                LET R5 :B= DSTMP0 ;GET MSD
      033546 116705 145002                                MOV      DSTMP0,R5
6638 033552                                IF SIGNO EQ #-1 THEN
      033552 026727 144672 177777                                CMP      SIGNO,# 1
      033560 001002                                BNE      $51376
6639 033562                                LET R5 := R5 + #10.
      033562 062705 000012                                ADD      #10.,R5
6640 033566                                ENDIF
      033566                                $51376:
6641 033566                                LET DSBUF1 :B= OVPNCH(R5)
      033566 116567 056330 030644                                MOV      OVPNCH(R5),DSBUF1
6642
6643
6644 033574                                CASE 4
      033574 000446                                BR       $51365
      033576                                $51367:
6645 033576                                LET DESCRO := DATA00 SET.BY #40000 ;TRAILING SEPARATE

```

ADDN

033576 016767 144760 145012  
 033604 052767 040000 145004  
 6646 033612  
 033612 016705 144664  
 033616 005205  
 6647 033620  
 033620 005767 144624  
 033624 001003  
 6648 033626  
 033626 112715 000053  
 6649 033632  
 033632 000402  
 033634  
 6650 033634  
 033634 112715 000055  
 6651 033640  
 033640  
 6652  
 6653 033640  
 033640 000424  
 033642  
 6654 033642  
 033642 016767 144714 144746  
 033650 052767 050000 144740  
 6655 033656  
 033656 016705 144620  
 033662 005205  
 6656 033664  
 033664 105015  
 6657 033666  
 033666 005767 144556  
 033672 001004  
 6658 033674  
 033674 112767 000053 030535  
 6659 033702  
 033702 000403  
 033704  
 6660 033704  
 033704 112767 000055 030525  
 6661 033712  
 033712  
 6662 033712  
 033712  
 6663  
 6664  
 6665 033712  
 033712 005037 000500  
 6666 033716  
 033716 005267 144556  
 6667 033722  
 033722 025727 144552 000005  
 033730 003402  
 6668 033732  
 033732 000167 001232  
 6669 033736  
 033736  
 6670

LET R5 := TEMP2 \* #1  
 IF SIGN0 EQ #0 THEN  
 LET (R5) :B= #53 ;+ SIGN  
 ELSE  
 LET (R5) :B= #55 ;- SIGN  
 ENDIF  
 CASE 5  
 LET DESCRO := DATA00 SET.BY #50000 ;LEADING SEPARATE  
 LET R5 := TEMP2 \* #1  
 LET (R5) :B= #0 ;RESTORE TRAILING SEPARATE  
 IF SIGN0 EQ #0 THEN  
 LET DSBUF1-1 :B= #53  
 ELSE  
 LET DSBUF1-1 :B= #55  
 ENDIF  
 ENDSELECT  
 ;INCR TEMP1 FROM #1 TO #5 BY #1 SRC2 DATA TYPES  
 INLINE <CLR @#TEMP1>  
 INLINE <INC2: INC TEMP1>  
 IF TEMP1 GT #5 THEN  
 INLINE <JMP END2>  
 ENDIF

MOV DATA00,DESCRO  
 BIS #40000,DESCRO  
 MOV TEMP2,R5  
 INC R5  
 TST SIGN0  
 BNE \$51377  
 MOVB #53,(R5)  
 BR \$51400  
 \$51377:  
 MOVB #55,(R5)  
 \$51400:  
 BR \$51365  
 \$51366:  
 MOV DATA00,DESCRO  
 BIS #50000,DESCRO  
 MOV TEMP2,R5  
 INC R5  
 CLRB (R5)  
 TST SIGN0  
 BNE \$51401  
 MOVB #53,DSBUF1 1  
 BR \$51402  
 \$51401:  
 MOVB #55,DSBUF1 1  
 \$51402:  
 \$51365:  
 CLR @#TEMP1  
 INC2: INC TEMP1  
 CMP TEMP1,#5  
 BLE \$51403  
 JMP END2  
 \$51403:

ADDN

```

6671                                     ;*
6672                                     ; RESTORE MSD & LSD OF DSBUF2
6673                                     ;
6674
6675 033736                               LET DSBUF2 :B= DSTMP2
        033736 116767 144614 030536
6676 033744                               LET TEMP3 := #DSBUF2-1 + DATA02
        033744 012767 064501 144532
        033752 066767 144610 144524
6677 033760                               LET @TEMP3 :B= DSTMP3
        033760 116777 144573 144516
6678
6679                                     ;*
6680                                     ; SIGN DSBUF2 BY DATA TYPE
6681                                     ;
6682                                     ;
6683 033766                               SELECT TEMP1 OF 5
        033766 016746 144506
        033772 006316
        033774 060716
        033776 063607
        034000
        034000 000012
        034002 000052
        034004 000116
        034006 000162
        034010 000226
6684 034012                               CASE 1
        034012
6685 034012                               LET DESCR2 := DATA02
        034012 016767 144550 144602
6686 034020                               IF SIGN1 EQ #0 THEN
        034020 005767 144426
        034024 001004
        034026                               LET @TEMP3 :B= @TEMP3 SET.BY #60 ;*
6687 034026 152777 000060 144450
        034026
6688 034034                               ELSE
        034034 000403
        034036
        034036                               LET @TEMP3 :B= @TEMP3 SET.BY #160;-
6689 034036 152777 000160 144440
        034036
6690 034044                               ENDIF
        034044
6691 034044                               LET DSBUF2-1 :B= #0
        034044 105067 030431
6692
6693 034050                               CASE 2
        034050 000512
        034052
6694 034052                               LET DESCR2 := DATA02 SET.BY #20000 ;TRAILING OVERPUNCH
        034052 016767 144510 144542
        034060 052767 020000 144534
6695 034066                               LET R5 :B= DSTMP3
        034066 116705 144465
6696 034072                               IF SIGN1 EQ #-1 THEN
        034072 026727 144354 177777
        034100 001002

```

```

MOV      DSTMP2,DSBUF2
MOV      #DSBUF2-1,TEMP3
ADD      DATA02,TEMP3
MOV      DSTMP3,@TEMP3

MOV      TEMP1,-(SP)
ASL      (SP)
ADD      PC,(SP)
ADD      @(SP)+,PC
$51404:
.WORD   $51412-$51404
.WORD   $51411-$51404
.WORD   $51410-$51404
.WORD   $51407-$51404
.WORD   $51406-$51404
$51412:
MOV      DATA02,DESCR2
TST      SIGN1
BNE      $51413
BISB     #60,@TEMP3
BR       $51414
$51413:
BISB     #160,@TEMP3
$51414:
CLRB     DSBUF2 1
BR       $51405
$51411:
MOV      DATA02,DESCR2
BIS      #20000,DESCR2
MOV      DSTMP3,R5
CMP      SIGN1,#1
BNE      $51415

```

ADDN

6697	034102				LET R5 := R5 + #10.		
	034102	062705	000012			ADD	#10.,R5
6698	034106				ENDIF		
	034106					\$51415:	
6699	034106				LET @TEMP3 :B= OVPNCH(R5)		
	034106	116577	056330	144370		MOVB	OVPNCH(R5),@TEMP3
6700							
6701	034114				CASE 3		
	034114	000470				BR	\$51405
	034116					\$51410:	
6702	034116				LET DESCR2 := DATA02 SET.BY #30000 ;LEADING OVERPUNCH		
	034116	016767	144444	144476		MOV	DATA02,DESCR2
	034124	052767	030000	144470		BIS	#30000,DESCR2
6703	034132				LET R5 :B= DSTMP2		
	034132	116705	144420			MOVB	DSTMP2,R5
6704	034136				IF SIGN1 EQ #-1 THEN		
	034136	026727	144310	177777		CMP	SIGN1,#-1
	034144	001002				BNE	\$51416
6705	034146				LET R5 := R5 + #10.		
	034146	062705	000012			ADD	#10.,R5
6706	034152				ENDIF		
	034152					\$51416:	
6707	034152				LET DSBUF2 :B= OVPNCH(R5)		
	034152	116567	056330	030322		MOVB	OVPNCH(R5),DSBUF2
6708							
6709	034160				CASE 4		
	034160	000446				BR	\$51405
	034162					\$51407:	
6710	034162				LET DESCR2 := DATA02 SET.BY #40000 ;TRAILING SEPARATE		
	034162	016767	144400	144432		MOV	DATA02,DESCR2
	034170	052767	040000	144424		BIS	#40000,DESCR2
6711	034176				LET R5 := TEMP3 + #1		
	034176	016705	144302			MOV	TEMP3,R5
	034202	005205				INC	R5
6712	034204				IF SIGN1 EQ #0 THEN		
	034204	005767	144242			TST	SIGN1
	034210	001003				BNE	\$51417
6713	034212				LET (R5) :B= #53		
	034212	112715	000053			MOVB	#53,(R5)
6714	034216				ELSE		
	034216	000402				BR	\$51420
	034220					\$51417:	
6715	034220				LET (R5) :B= #55		
	034220	112715	000055			MOVB	#55,(R5)
6716	034224				ENDIF		
	034224					\$51420:	
6717							
6718	034224				CASE 5		
	034224	000424				BR	\$51405
	034226					\$51406:	
6719	034226				LET DESCR2 := DATA02 SET.BY #50000 ;LEADING SEPARATE		
	034226	016767	144334	144366		MOV	DATA02,DESCR2
	034234	052767	050000	144360		BIS	#50000,DESCR2
6720	034242				LET R5 := TEMP3 + #1		
	034242	016705	144236			MOV	TEMP3,R5
	034246	005205				INC	R5
6721	034250				LET (R5) :B= #0		



ADDN

6722	034250	105015							
	034252	005767	144174		IF SIGN1 EQ #0 THEN		CLRB	(R5)	
	034256	001004					TST	SIGN1	
6723	034260				LET DSBUF2-1 :B= #53		BNE	\$51421	
	034266	112767	000053	030213			MOVB	#53,DSBUF2-1	
6724	034266				ELSE		BR	\$51422	
	034270	000403					\$51421:		
6725	034270				LET DSBUF2-1 :B= #55		MOVB	#55,DSBUF2 1	
	034276	112767	000055	030203			\$51422:		
6726	034276				ENDIF		\$51405:		
	034276				ENDSELECT				
6727	034276								
	034276								
6728									
6729					;INCR DATTYP FROM #1 TO #5 BY #1 ;DS.EXP LOOP				
6730	034276				INLINE <CLR @DATTYP>				
	034276	005037	000464				CLR @DATTYP		
6731	034302				INLINE <INC3: INC DATTYP>		INC3: INC DATTYP		
	034302	005267	144156						
6732	034306				IF DATTYP GT #5 THEN		CMP	DATTYP,#5	
	034306	026727	144152	000005			BLE	\$51423	
	034314	003402							
6733	034316				INLINE <JMP END3>		JMP END3		
	034316	000167	000642						
6734	034322				ENDIF				
	034322						\$51423:		
6735									
6736					;*				
6737					; RESTORE MSD & LSD OF DS.EXP				
6738					;-				
6739									
6740	034322				LET DS.EXP :B= DS'TMP4 ;MSD				
	034322	116767	144232	030256			MOVB	DSTMP4,DS.EXP	
6741	034330				LET TEMP4 := #DS.EXP-1 + DATA04		MOV	#DS.EXP 1,TEMP4	
	034330	012767	064605	144150			ADD	DATA04,TEMP4	
	034336	066767	144230	144142					
6742	034344				LET @TEMP4 :B= DSTMP5 ;LSD		MOVB	DSTMP5,@TEMP4	
	034344	116777	144211	144134					
6743									
6744					;*				
6745					; SIGN DS.EXP BY DATA TYPE				
6746					;-				
6747									
6748	034352				SELECT DATTYP OF 5				
	034352	016746	144106				MOV	DATTYP, (SP)	
	034356	006316					ASL	(SP)	
	034360	060716					ADD	PC,(SP)	
	034362	063607					ADD	@(SP)+,PC	
	034364						\$51424:		
	034364	000012					.WORD	\$51432-\$51424	
	034366	000060					.WORD	\$51431-\$51424	
	034370	000136					.WORD	\$51430-\$51424	
	034372	000214					.WORD	\$51427-\$51424	
	034374	000260					.WORD	\$51426-\$51424	
6749	034376				CASE 1				

ADDN

6750	034376				LET DESCR4 := DATA04	;SIGNED ZONED	\$51432:	
	034376	016767	144170	144222				MOV DATA04,DESCR4
6751	034404				IF DATA04 NE #0 THEN			TST DATA04
	034404	005767	144162					BEQ \$51433
	034410	001414						
6752	034412				IF SIGN2 EQ #0 THEN			TST SIGN2
	034412	005767	144036					BNE \$51434
	034416	001004						
6753	034420				LET @TEMP4 :B= @TEMP4 SET.BY #60			BISB #60,@TEMP4
	034420	152777	000060	144060				BR \$51435
6754	034426				ELSE			
	034426	000403						
	034430							
6755	034430				LET @TEMP4 :B= @TEMP4 SET.BY #160		\$51434:	
	034430	152777	000160	144050				BISB #160,@TEMP4
6756	034436				ENDIF			
	034436							
6757	034436				LET DS.EXP 1 :B= #0		\$51435:	
	034436	105067	030143					CLRB DS.EXP-1
6758	034442				ENDIF			
	034442							
6759								
6760	034442				CASE 2			BR \$51425
	034442	000524						
	034444							
6761	034444				LET DESCR4 := DATA04 SET.BY #20000	;TRAILING OVERPUNCH	\$51431:	
	034444	016767	144122	144154				MOV DATA04,DESCR4
	034452	052767	020000	144146				BIS #20000,DESCR4
6762	034460				IF DATA04 NE #0 THEN			
	034460	005767	144106					TST DATA04
	034464	001415						BEQ \$51436
6763	034466				LET R5 :B= DSTMP5			MOVB DSTMP5,R5
	034466	116705	144067					SUB #60,R5
6764	034472				LET R5 := R5 - #60			
	034472	162705	000060					
6765	034476				IF SIGN2 EQ #-1 THEN			CMP SIGN2,# 1
	034476	026727	143752	177777				BNE \$51437
	034504	001002						
6766	034506				LET R5 := R5 * #10.			ADD #10.,R5
	034506	062705	000012					
6767	034512				ENDIF			
	034512							
6768	034512				LET @TEMP4 :B= OVPNCH(R5)		\$51437:	
	034512	116577	056330	143766				MOVB OVPNCH(R5),@TEMP4
6769	034520				ENDIF			
	034520							
6770								
6771	034520				CASE 3			BR \$51425
	034520	000475						
	034522							
6772	034522				LET DESCR4 := DATA04 SET.BY #30000	;LEADING OVERPUNCH	\$51430:	
	034522	016767	144044	144076				MOV DATA04,DESCR4
	034530	052767	030000	144070				BIS #30000,DESCR4
6773	034536				IF DATA04 NE #0 THEN			
	034536	005767	144030					TST DATA04
	034542	001415						BEQ \$51440

ADDN

6774	034544				LET R5 :B= DSTMP4		
	034544	116705	144010			MOV B	DSTMP4,R5
6775	034550				LET R5 := R5 #60		
	034550	162705	000060			SUB	#60,R5
6776	034554				IF SIGN2 EQ # 1 THEN		
	034554	026727	143674	177777		CMP	SIGN2,#-1
	034562	001002				BNE	\$51441
6777	034564				LET R5 := R5 + #10.		
	034564	062705	000012			ADD	#10.,R5
6778	034570				ENDIF		
	034570					\$51441:	
6779	034570				LET DS.EXP :B OVPNCH(R5)		
	034570	116567	056330	030010		MOV B	OVPNCH(R5),DS.EXP
6780	034576				ENDIF		
	034576					\$51440:	
6781							
6782	034576				CASE 4		
	034576	000446				BR	\$51425
	034600					\$51427:	
6783	034600				LET DESCR4 :- DATA04 SET.BY #40000 ;TRAILING SEPARATE		
	034600	016767	143766	144020		MOV	DATA04,DESCR4
	034606	052767	040000	144012		BIS	#40000,DESCR4
6784	034614				LET R5 := TEMP4 + #1		
	034614	016705	143666			MOV	TEMP4,R5
	034620	005205				INC	R5
6785	034622				IF SIGN2 EQ #0 THEN		
	034622	005767	143626			TST	SIGN2
	034626	001003				BNE	\$51442
6786	034630				LET (R5) :B= #53		
	034630	112715	000053			MOV B	#53,(R5)
6787	034634				ELSE		
	034634	000402				BR	\$51443
	034636					\$51442:	
6788	034636				LET (R5) :B= #55		
	034636	112715	000055			MOV B	#55,(R5)
6789	034642				ENDIF		
	034642					\$51443:	
6790							
6791	034642				CASE 5		
	034642	000424				BR	\$51425
	034644					\$51426:	
6792	034644				LET DESCR4 := DATA04 SET.BY #50000		
	034644	016767	143722	143754		MOV	DATA04,DESCR4
	034652	052767	050000	143746		BIS	#50000,DESCR4
6793	034660				LET R5 := TEMP4 + #1		
	034660	016705	143622			MOV	TEMP4,R5
	034664	005205				INC	R5
6794	034666				LET (R5) :B= #0		
	034666	105015				CLRB	(R5)
6795	034670				IF SIGN2 EQ #0 THEN		
	034670	005767	143560			TST	SIGN2
	034674	001004				BNE	\$51444
6796	034676				LET DS.EXP 1 :B= #53		
	034676	112767	000053	027701		MOV B	#53,DS.EXP 1
6797	034704				ELSE		
	034704	000403				BR	\$51445
	034706					\$51444:	

ADDN

6798	034706				LET DS.EXP-1 :B= #55		
	034706	112767	000055	027671		MOVB	#55,DS.EXP 1
6799	034714				ENDIF		
	034714					\$51445:	
6800	034714				ENDSELECT		
	034714					\$51425:	
6801							
6802							
6803					:+		
6804					:		
6805					:-		
6806	034714				LET R0 :- #DS.DST-1		
	034714	012700	064543			MOV	#DS.DST-1,R0
6807	034720				LET R1 := #41		
	034720	012701	000041			MOV	#41,R1
6808	034724				INLINE <DOIT1: CLR8 (R0)+>		
	034724	105020				DOIT1:	CLR8 (R0).
6809	034726				INLINE <SOB R1, DOIT1>		
	034726	077102				SOB	R1, DOIT1
6810	034730				LET R0 := DESCRO		
	034730	016700	143662			MOV	DESCRO,R0
6811	034734				LET R1 := DESCR1		
	034734	016701	143660			MOV	DESCR1,R1
6812	034740				LET R2 := DESCR2		
	034740	016702	143656			MOV	DESCR2,R2
6813	034744				LET R3 := DESCR3		
	034744	016703	143654			MOV	DESCR3,R3
6814	034750				LET R4 := DESCR4		
	034750	016704	143652			MOV	DESCR4,R4
6815	034754				LET R5 := DESCR5		
	034754	016705	143650			MOV	DESCR5,R5
6816	034760				LET R4.EXP := R4		
	034760	010467	143550			MOV	R4,R4.EXP
6817	034764				LET R5.EXP := R5		
	034764	010567	143546			MOV	R5,R5.EXP
6818	034770				INLINE <ADDN>		
	034770	076050				ADDN	
6819							
6820					:+		
6821					:		
6822					:-		
6823							
6824	034772				LET CC := PSW		
	034772	016767	143000	143520		MOV	PSW,CC
6825	035000				LET CC := CC CLR.BY #177760		
	035000	042767	177760	143512		BIC	#177760,CC
6826	035006				IF CC NE CC.EXP THEN		
	035006	026767	143506	143506		CMP	CC,CC.EXP
	035014	001402				BEQ	\$51446
6827	035016				CALL ERROR		
	035016	004767	022656			JSR	PC,ERROR
6828	035022				ENDIF		
	035022					\$51446:	
6829	035022				IF R0 NE R0.EXP THEN		
	035022	020067	143476			CMP	R0,R0.EXP
	035026	001402				BEQ	\$51447
6830	035030				CALL ERROR		

ADDN

6831	035030	004767	022644		JSR	PC.ERROR
	035034			ENDIF		
6832	035034			IF R1 NE R1.EXP THEN	\$51447:	
	035034	020167	143466			
	035040	001402		CALL ERROR	CMP	R1,R1.EXP
6833	035042				BEQ	\$51450
	035042	004767	022632	ENDIF	JSR	PC.ERROR
6834	035046					
	035046			IF R2 NE R2.EXP THEN	\$51450:	
6835	035046	020267	143456			
	035052	001402		CALL ERROR	CMP	R2,R2.EXP
6836	035054				BEQ	\$51451
	035054	004767	022620	ENDIF	JSR	PC.ERROR
6837	035060					
	035060			IF R3 NE R3.EXP THEN	\$51451:	
6838	035060	020367	143446			
	035064	001402		CALL ERROR	CMP	R3,R3.EXP
6839	035066				BEQ	\$51452
	035066	004767	022606	ENDIF	JSR	PC.ERROR
6840	035072					
	035072			IF R4 NE R4.EXP THEN	\$51452:	
6841	035072	020467	143436			
	035076	001402		CALL ERROR	CMP	R4,R4.EXP
6842	035100				BEQ	\$51453
	035100	004767	022574	ENDIF	JSR	PC.ERROR
6843	035104					
	035104			IF R5 NE R5.EXP THEN	\$51453:	
6844	035104	020567	143426			
	035110	001402		CALL ERROR	CMP	R5,R5.EXP
6845	035112				BEQ	\$51454
	035112	004767	022562	ENDIF	JSR	PC.ERROR
6846	035116					
	035116				\$51454:	
6847						
6848						
6849				:+		
6850				:	CHECK RESULTS VS EXPECTED RESULTS	
6851				:		
6852	035116			LET R0 := #DS.DST-1		
	035116	012700	064543		MOV	#DS.DST-1,R0
6853	035122			LET R1 := #DS.EXP 1		
	035122	012701	064605		MOV	#DS.EXP-1,R1
6854	035126			LET ERRFLG := #0		
	035126	005067	143324		CLR	ERRFLG
6855	035132			WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO		
	035132				\$51455:	
	035132	020027	064604		CMP	R0,#DS.DST+40
	035136	001410			BEQ	\$51456
	035140	005767	143312		TST	ERRFLG
	035144	001005			BNE	\$51456
6856	035146			IFB (R0)+ NE (R1)+ THEN		
	035146	122021			CMPB	(R0)+,(R1)+

```

ADDN
6857 035150 001402          CALL ERROR          BEQ      $51457
        035152          CALL ERROR          JSR      PC,ERROR
6858 035152 004767 022522          ENDIF
        035156          ENDIF          $51457:
6859 035156 000765          ENDDO          $51456: BR      $51455
        035160
6860          ;ENDINC
6861 035160 000167 177116          INLINE <JMP INC3>          JMP INC3
        035164          INLINE <END3:>          END3:
6862 035164
6863          ;ENDINC
6864 035164 000167 176526          INLINE <JMP INC2>          JMP INC2
        035170          INLINE <END2:>          END2:
6865 035170
6866          ;ENDINC
6867 035170 000167 176136          INLINE <JMP INC1>          JMP INC1
        035174          INLINE <END1:>          END1:
6868 035174
6869 035174 062767 000022 143342          LET TSTCAS := TSTCAS + #22          ADD      #22,TSTCAS
6870 035202 005367 143262          LET COUNT := COUNT - #1          DEC      COUNT
        035206          ;UNTIL COUNT EQ #0
6871 035206 005767 143256          IF COUNT NE #0 THEN          TST      COUNT
6872 035212 001402          ;UNTIL COUNT EQ #0          BEQ      $51460
        035214          ;UNTIL COUNT EQ #0          JMP $5050
6873 035214 000167 175544          INLINE <JMP $5050>
6874 035220          ENDIF          $51460:
        035220
6875
6876          ;+
6877          ;          INLINE CASE - ADDNI, USING UNSIGNED ZONED DATA
6878          ;-
6879
6880 035220 004767 021450          CALL CLRDSB          JSR      PC,CLRDSB
        035224
6881
6882          ;+
6883          ;          LOAD DSBUF1
6884          ;-
6885
6886 035224 012700 065430          LET R0 := #ADD1+2          ;SOURCE
        035230          LET R1 := TADDI CLR.BY #10000          ;# OF BYTES TO XFER
6887 035230 016701 030666          ;DESTINATION
        035234 042701 010000          MOV      #ADD1+2,R0
        035240          LET R2 :- #DSBUF1
        035244          MOV      TADDI,R1
6888 035240 012702 064440          MOV      #10000,R1
        035244          MOV      #DSBUF1,R2
6889 035244 112022          MOV      #DSBUF1,R2
        035244          MOV      #DSBUF1,R2
        1$: MOV      (R0)+, (R2)+
6890 035246          INLINE <          SOB      R1,      1$>

```

ADDN

```

035246 077102                                SOB      R1,      1$
6891
6892
6893      ;+                                LOAD DSBUF2
6894      ;-
6895
6896 035250                                LET R0 := #ADD1A+2
035250 012700 065452                                MOV      #ADD1A+2,R0
6897 035254                                LET R1 := TADDI+4 CLR.BY #10000
035254 016701 030646                                MOV      TADDI+4,R1
035260 042701 010000                                BIC      #10000,R1
6898 035264                                LET R2 := #DSBUF2
035264 012702 064502                                MOV      #DSBUF2,R2
6899 035270                                INLINE   <2$:  MOVB   (R0)+, (R2)+>
035270 112022                                2$:     MOVB   (R0)+, (R2)

6900 035272                                INLINE   <      SOB   R1,      2$>
035272 077102                                SOB      R1,      2$
6901
6902      ;+                                LOAD DS.EXP
6903      ;-
6904
6905
6906 035274                                LET R0 := #ADD1B+2
035274 012700 065476                                MOV      #ADD1B+2,R0
6907 035300                                LET R1 := TADDI+10 CLR.BY #10000
035300 016701 030626                                MOV      TADDI+10,R1
035304 042701 010000                                BIC      #10000,R1
6908 035310                                LET R2 := #DS.EXP
035310 012702 064606                                MOV      #DS.EXP,R2
6909 035314                                INLINE   <3$:  MOVB   (R0)+, (R2)>
035314 112012                                3$:     MOVB   (R0)+, (R2)
6910 035316                                INLINE   <      BISB  #60,   (R2)+>
035316 152722 000060                                BISB    #60,   (R2)

6911 035322                                INLINE   <      SOB   R1,      3$>
035322 077104                                SOB      R1,      3$
6912
6913 035324                                CALL CLRREG                                ;ENSURE R0-R5 ARE CLR
035324 004767 021110                                JSR      PC,CLRREG
6914
6915 035330                                INLINE <ADDNI>                                ADDNI
035330 076150
6916 035332                                INLINE <.WORD TADDI>                                .WORD TADDI
035332 066122
6917 035334                                INLINE <.WORD TADDI+4>                                .WORD TADDI+4
035334 066126
6918 035336                                INLINE <.WORD TADDI+10>                                .WORD TADDI+10
035336 066132
6919 035340                                LET CC := PSW
035340 016767 142432 143152                                MOV      PSW,CC
6920 035346                                LET CC := CC CLR.BY #177760
035346 042767 177760 143144                                BIC      #177760,CC
6921 035354                                IF CC NE #0 THEN
035354 005767 143140                                TST     CC
035360 001402                                BEQ     $51461
6922 035362                                CALL ERROR
035362 004767 022312                                JSR     PC,ERROR
6923 035366                                ENDIF

```

ADDN

\$51461:

6924 035366

6925

6926

6927

6928

6929

035366

035366

035370

035372

035374

035376

035400

035402

6930

035402

6931

035406

6932

035406

035406

035410

035412

035414

035416

035420

035422

6933

035422

6934

035426

6935

6936

6937

6938

6939

6940

035426

6941

035432

6942

035432

6943

035436

035442

035442

035446

035450

035454

6944

035456

035456

6945

035462

6946

035466

6947

035466

035470

:+  
;  
:-

ENSURE GPR'S STILL ZERO

IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN

005700

001004

005701

001002

005702

001402

CALL ERROR

ENDIF

IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN

005703

001004

005704

001002

005705

001402

CALL ERROR

ENDIF

:+  
;  
:-

CHECK RESULTS

LET R0 := #DS.DST

LET R1 := #DS.EXP

LET ERRFLG := #0

WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO

020027 064604

001410

005767 143002

001005

IFB (R0)+ NE (R1)+ THEN

CALL ERROR

ENDIF

ENDDO

TST R0  
BNE \$51462  
TST R1  
BNE \$51462  
TST R2  
BEQ \$51463

\$51462:

JSR PC,ERROR

\$51463:

TST R3  
BNE \$51464  
TST R4  
BNE \$51464  
TST R5  
BEQ \$51465

\$51464:

JSR PC,ERROR

\$51465:

MOV #DS.DST,R0  
MOV #DS.EXP,R1  
CLR ERRFLG

\$51466:

CMP R0,#DS.DST+40  
BEQ \$51467  
TST ERRFLG  
BNE \$51467

CMPB (R0)+,(R1)+  
BEQ \$51470

JSR PC,ERROR

\$51470:

BR \$51466

\$51467:



ADDN

6948  
6949

SUBN

```

6951 .SBTTL SUBN
6952 ;*****
6953 ;
6954 ; SUBN TEST
6955 ;
6956 ;*****
6957
6958 035470 ROUTINE XSUBN
6959 035470 XSUBN:
6960 035470 000004 INLINE <SCOPE> SCOPE
6961 035472 000240 INLINE <NOP> NOP
6962 035474 012767 000010 142702 LET $TIMES := #10 ;ITERATION COUNT MOV #10,$TIMES
6963 035502 016767 027542 142760 LET COUNT := TADD ;# OF TEST CASES MOV TADD,COUNT
6964 035510 012767 065252 143026 LET TSTCAS := #TADD+2 ;POINT TO FIRST TEST CASE MOV #TADD-2,TSTCAS
6965
6966 ;+ CLR R0.EXP TO R3.EXP
6967 ;
6968 ;
6969 035516 INCR R0 FROM #0 TO #6 BY #2
6970 035516 005000 CLR R0
6971 035520 000402 BR $51473
6972 035522 062700 000002 $51474: ADD #2,R0
6973 035526 020027 000006 $51475: CMF R0,#6
6974 035532 003003 BG1 $51475
6975 035534 INLINE <CLR R0.EXP(R0)> CLR R0.EXP(R0)
6976 035534 005060 000524 ENDINC BR $51474
6977 035540 000770 $51475:
6978 035542 ;REPEAT OUTER LOOP FOR TEST CASES
6979 ;
6980 035542 004767 021126 CALL CLRDSB ;CLR DS BUFFERS JSR PC,CLRDSB
6981 ;+
6982 ; LOAD DATA00 AND CC.EXP FROM TEST CASE
6983 ;
6984 035546 LET R0 := TSTCAS MOV TSTCAS,R0
6985 035546 016700 142772 LET DATA00 := (R0)+ ;SRC1 LENGTH MOV (R0)+,DATA00
6986 035552 012067 143004 LET DATA01 := (R0)+ ;POINTER TO SRC1 DATA MOV (R0)+,DATA01
6987 035556 012067 143002 LET DATA02 := (R0)+ ;SRC2 LENGTH MOV (R0)+,DATA02
6988 035562 012067 143000 LET DATA03 := (R0)+ ;POINTER TO SRC2 DATA MOV (R0)+,DATA03
6989 035566 012067 142776

```

SUBN

```

6985 035572          LET DATA04 := (R0)+          ;DEST LENGTH
      035572 012067 142774
6986 035576          LET DATA05 := (R0)+          ;POINTER TO DS.EXP DATA (SUM)
      035576 012067 142772
6987 035602          LET CC.EXP := (R0)+          ;CC.EXP (SUM)
      035602 012067 142714
6988 035606          LET DATA05 := (R0)+          ;POINTER TO DS.EXP DATA (DIFF)
      035606 012067 142762
6989 035612          LET CC.EXP := (R0)+          ;CC.EXP (DIFF)
      035612 012067 142704
6990
6991                ;+
6992                ;   LOAD DSBUF1
6993                ;-
6994
6 75 035616          IF DATA00 NE #0 THEN          ;CHECK FOR ZERO LENGTH BUFFER
      035616 005767 142740
      035622 001413
6996 035624          LET R0 := DATA01
      035624 016700 142734
6997 035630          LET SIGN0 := (R0)+          ;STORE SIGN AND POINT AT FIRST BYTE
      035630 012067 142614
6998 035634          LET R1 := DATA00          ;# OF BYTES TO TRANSFER
      035634 016701 142722
6999 035640          LET R2 := #DSBUF1          ;TRANSFER DESTINATION
      035640 012702 064440
7000 035644          INLINE <1$:>
      035644
7001 035644          INLINE <MOVB (R0)+,(R2)+>
      035644 112022
7002 035646          INLINE <SOB R1,1$>
      035646 077102
7003 035650          ELSE
      035650 000402
      035652
7004 035652          LET SIGN0 := #0
      035652 005067 142572
7005 035656          ENDIF
      035656
7006
7007                ;+
7008                ;   LOAD DSBUF2
7009                ;-
7010
7011 035656          IF DATA02 NE #0 THEN
      035656 005767 142704
      035662 001413
7012 035664          LET R0 := DATA03
      035664 016700 142700
7013 035670          LET SIGN1 := (R0)+
      035670 012067 142556
7014 035674          LET R1 := DATA02
      035674 016701 142666
7015 035700          LET R2 := #DSBUF2
      035700 012702 064502
7016 035704          INLINE <2$:>
      035704

```

```

MOV      (R0)+,DATA04
MOV      (R0)+,DATA05
MOV      (R0)+,CC.EXP
MOV      (R0)+,DATA05
MOV      (R0)+,CC.EXP

```

```

TST      DATA00
BEQ      $51476
MOV      DATA01,R0
MOV      (R0)+,SIGN0
MOV      DATA00,R1
MOV      #DSBUF1,R2
1$:
MOVB (R0)+,(R2)+
SOB R1,1$
BR      $51477
$51476:
CLR      SIGN0
$51477:

```

```

TST      DATA02
BEQ      $51500
MOV      DATA03,R0
MOV      (R0)+,SIGN1
MOV      DATA02,R1
MOV      #DSBUF2,R2
2$:

```

SUBN

7017	035704			INLINE <MOVB (R0)+,(R2)+>				MOV B (R0)+,(R2)+
	035704	112022						
7018	035706			INLINE <SOB R1,2\$>				SOB R1,2\$
	035706	077102						
7019	035710			ELSE				BR \$51501
	035710	000402						
	035712							\$51500:
7020	035712			LET SIGN1 := #0				CLR SIGN1
	035712	005067	142534					
7021	035716			ENDIF				\$51501:
	035716							
7022								
7023				;;				
7024				LOAD DS.EXP				
7025				;-				
7026								
7027	035716			IF DATA04 NE #0 THEN				
	035716	005767	142650					TST DATA04
	035722	001415						BEQ \$51502
7028	035724			LET R0 := DATA05				MOV DATA05,R0
	035724	016700	142644					
7029	035730			LET SIGN2 := (R0)+				MOV (R0)+,SIGN2
	035730	012067	142520					
7030	035734			LET R1 := DATA04				MOV DATA04,R1
	035734	016701	142632					
7031	035740			LET R2 := #DS.EXP				MOV #DS.EXP,R2
	035740	012702	064606					
7032	035744			INLINE <3\$:>				3\$:
	035744							
7033	035744			INLINE <MOVB (R0)+,(R2)+>				MOV B (R0)+,(R2)+
	035744	112012						
7034	035746			INLINE <BISB #60, (R2)+>				BISB #60, (R2)+
	035746	152722	000060					
7035	035752			INLINE <SOB R1,3\$>				SOB R1,3\$
	035752	077104						
7036	035754			ELSE				BR \$51503
	035754	000402						
	035756							\$51502:
7037	035756			LET SIGN2 := #0				CLR SIGN2
	035756	005067	142472					
7038	035762			ENDIF				\$51503:
	035762							
7039								
7040				;;				
7041				LOAD DSTMP'S WITH MDS & LSD OF EACH BUFFER				
7042				;				
7043				;				
7044	035762			LET DSTMP0 :B= DSBUF1				MOV B DSBUF1,DSTMP0
	035762	116767	026452	142564				
7045	035770			LET TEMPO := #DSBUF1-1 + DATA00				MOV #DSBUF1-1,TEMPO
	035770	012767	064437	142500				ADD DATA00,TEMPO
	035776	066767	142560	142472				
7046	036004			LET DSTMP1 :B= @TEMPO				MOV B @TEMPO,DSTMP1
	036004	117767	142466	142543				
7047	036012			LET DSTMP2 :B= DSBUF2				MOV B DSBUF2,DSTMP2
	036012	116767	026464	142536				
7048	036020			LET TEMPO := #DSBUF2-1 + DATA02				

SUBN

036020	012767	064501	142450			MOV	#DSBUF2-1,TEMPO
036026	066767	142534	142442			ADD	DATA02,TEMPO
7049 036034				LET DSTMP3 :B= @TEMPO		MOVB	@TEMPO,DSTMP3
036034	117767	142436	142515			MOVB	DS.EXP,DSTMP4
7050 036042				LET DSTMP4 :B= DS.EXP		MOV	#DS.EXP-1,TEMPO
036042	116767	026540	142510			ADD	DATA04,TEMPO
7051 036050				LET TEMPO := #DS.EXP-1 + DATA04		MOVB	@TEMPO,DSTMP5
036050	012767	064605	142420			MOV	#DSBUF1,DESCR1
036056	066767	142510	142412			MOV	#DSBUF2,DESCR3
7052 036064				LET DSTMP5 :B= @TEMPO		MOV	#DS.DST,DESCR5
036064	117767	142406	142467				
7053 036072				LET DESCR1 := #DSBUF1			
036072	012767	064440	142520				
7054 036100				LET DESCR3 := #DSBUF2			
036100	012767	064502	142516				
7055 036106				LET DESCR5 := #DS.DST			
036106	012767	064544	142514				
7056							
7057				; INCR TEMPO FROM #1 TO #5 BY #1	SRC1 DATA TYPE LOOP		
7058 036114				INLINE <CLR @TEMPO>		CLR	@TEMPO
036114	005037	000476				INC21:	INC TEMPO
7059 036120				INLINE <INC21: INC TEMPO>			
036120	005267	142352					
7060 036124				IF TEMPO GT #5 THEN		CMP	TEMPO,#5
036124	026727	142346	000005			BLE	\$51504
036132	003402						
7061 036134				INLINE <JMP END21>		JMP	END21
036134	000167	001626					
7062 036140				ENDIF			
036140						\$51504:	
7063							
7064				;+ RESTORE MSD & LSD OF DSBUF1			
7065				; -			
7066							
7067							
7068 036140				LET DSBUF1 :B= DSTMPO ;MSD		MOVB	DSTMPO,DSBUF1
036140	116767	142410	026272			MOV	#DSBUF1-1,TEMP2
7069 036146				LET TEMP2 := #DSBUF1-1 + DATA00		ADD	DATA00,TEMP2
036146	012767	064437	142326			MOVB	DSTMP1,@TEMP2
036154	066767	142402	142320				
7070 036162				LET @TEMP2 :B= DSTMP1 ;LSD			
036162	116777	142367	142312				
7071							
7072				;+ SIGN DSBUF1 BY DATA TYPE			
7073				; -			
7074							
7075							
7076 036170				SELECT TEMPO OF 5		MOV	TEMPO,(SP)
036170	016746	142302				ASL	(SP)
036174	006316					ADD	PC,(SP)
036176	060713					ADD	@(SP),PC
036200	063607						
036202						\$51505:	
036202	000012					.WORD	\$51513 \$51505
036204	000052					.WORD	\$51512 \$51505
036206	000116					.WORD	\$51511 \$51505
036210	000162					.WORD	\$51510 \$51505

SUBN

7077	036212	000226			CASE 1	.WORD	\$51507-\$51505
	036214					\$51513:	
7078	036214				LET DESCRO := DATA00 ;SIGNED ZONED	MOV	DATA00,DESCRO
	036214	016767	142342	142374			
7079	036222				IF SIGNO EQ #0 THEN	TST	SIGNO
	036222	005767	142222			BNE	\$51514
	036226	001004			LET @TEMP2 :B= @TEMP2 SET.BY #60 ;+		
7080	036230				ELSE	BISB	#60,@TEMP2
	036230	152777	000060	142244			
7081	036236				LET @TEMP2 :B= @TEMP2 SET.BY #160 ;-	BR	\$51515
	036236	000403			ENDIF	\$51514:	
	036240				LET DSBUF1-1 :B= #0 ;FIX LEADING SEPARATE RESULT	BISB	#160,@TEMP2
7082	036240	152777	000160	142234			
7083	036246				LET DSBUF1-1 :B= #0 ;FIX LEADING SEPARATE RESULT	CLR	DSBUF1 1
	036246	105067	026165				
7085					CASE 2		
7086	036252	000512				BR	\$51506
	036252					\$51512:	
7087	036254				LET DESCRO := DATA00 SET.BY #20000 ;TRAILING OVERPUNCH	MOV	DATA00,DESCRO
	036254	016767	142302	142334		BIS	#20000,DESCRO
	036262	052767	020000	142326			
7088	036270				LET R5 :B= DSTMP1 ;GET LSD	MOVB	DSTMP1,R5
	036270	116705	142261		IF SIGNO EQ #-1 THEN		
7089	036274					CMP	SIGNO,# 1
	036274	026727	142150	177777		BNE	\$51516
	036302	001002			LET R5 := R5 + #10. ;CHANGE INDEX FOR	ADD	#10.,R5
7090	036304	062705	000012		ENDIF		
7091	036310				LET @TEMP2 :B= OVPNCH(R5)	\$51516:	
	036310					MOVB	OVPNCH(R5),@TEMP2
7092	036310	116577	056330	142164			
7093					CASE 3		
7094	036316	000470				BR	\$51506
	036316					\$51511:	
7095	036320				LET DESCRO := DATA00 SET.BY #30000 ;LEADING OVERPUNCH	MOV	DATA00,DESCRO
	036320	016767	142236	142270		BIS	#30000,DESCRO
	036326	052767	030000	142262			
7096	036334				LET R5 :B= DSTMP0 ;GET MSD	MOVB	DSTMP0,R5
	036334	116705	142214		IF SIGNO EQ #-1 THEN		
7097	036340					CMP	SIGNO,# 1
	036340	026727	142104	177777		BNE	\$51517
	036346	001002			LET R5 := R5 + #10.	ADD	#10.,R5
7098	036350	062705	000012		ENDIF		
7099	036354				LET DSBUF1 :B= OVPNCH(R5)	\$51517:	
	036354					MOVB	OVPNCH(R5),DSBUF1
7100	036354	116567	056330	026056			
7101							
7102							

SUBN

```

7103 036362          CASE 4
      036362 000446
7104 036364          LET DESCRO := DATA00 SET.BY #40000 ;TRAILING SEPARATE
      036364 016767 142172 142224          $51510: BR $51506
      036372 052767 040000 142216          MOV DATA00,DESCRO
7105 036400          LET R5 := TEMP2 * #1          BIS #40000,DESCRO
      036400 016705 142076          MOV TEMP2,R5
      036404 005205          INC R5
7106 036406          IF SIGNC EQ #0 THEN
      036406 005767 142036
      036412 001003
7107 036414          LET (R5) :B= #53 ;+ SIGN          TST SIGN0
      036414 112715 000053          ;+ SIGN          BNE $51520
7108 036420          ELSE
      036420 000402
7109 036422          LET (R5) :B= #55 ;- SIGN          $51520: MOVB #53,(R5)
      036422 112715 000055          ;- SIGN          BR $51521
7110 036426          ENDIF          $51521:
      036426
7111
7112 036426          CASE 5
      036426 000424
7113 036430          LET DESCRO := DATA00 SET.BY #50000 ;LEADING SEPARATE          $51507: BR $51506
      036430 016767 142126 142160          ;LEADING SEPARATE          MOV DATA00,DESCRO
      036436 052767 050000 142152          BIS #50000,DESCRO
7114 036444          LET R5 := TEMP2 * #1          MOV TEMP2,R5
      036444 016705 142032          INC R5
7115 036452          LET (R5) :B= #0 ;RESTORE TRAILING SEPARATE          CLR (R5)
      036452 105015          IF SIGN0 EQ #0 THEN          TST SIGN0
7116 036454          ;RESTORE TRAILING SEPARATE          BNE $51522
      036454 005767 141770
      036460 001004
7117 036462          LET DSBUF1-1 :B= #53          MOVB #53,DSBUF1 1
      036462 112767 000053 025747          ELSE          BR $51523
7118 036470          $51522:
      036470 000403          LET DSBUF1-1 :B= #55          MOVB #55,DSBUF1 1
7119 036472          ENDIF          $51523:
      036472 112767 000055 025737          $51506:
7120 036500          ENDSELECT
      036500
7121 036500          ;INCR TEMP1 FROM #1 TO #5 BY #1          SRC2 DATA TYPES
      036500          INLINE <CLR @TEMP1>          CLR @TEMP1
7122
7123
7124 036500          ;INCR TEMP1 FROM #1 TO #5 BY #1          SRC2 DATA TYPES
      036500 005037 000500          INLINE <INC22: INC TEMP1>          INC22: INC TEMP1
7125 036504          IF TEMP1 GT #5 THEN
      036504 005267 141770
7126 036510          CMP TEMP1,#5
      036510 026727 141764 000005          BLE $51524
      036516 003402
7127 036520          INLINE <JMP END22>

```

SUBN

```

7128 036520 000167 001236                               JMP END22
      036524                                     ENDIF
7129 036524 012703 064502                               $51524:
      036524                                     MOV     #DSBUF2,R3
7130
7131                                     ;*
7132                                     ; RESTORE MSD & LSD OF DSBUF2
7133                                     ;
7134
7135 036530                                     LET DSBUF2 :B= DSTMP2
      036530 116767 142022 025744                       MOV     DSTMP2,DSBUF2
7136 036536                                     LET TEMP3 := #DSBUF2 1 + DATA02
      036536 012767 064501 141740                       MOV     #DSBUF2 1,TEMP3
      036544 066767 142016 141732                       ADD     DATA02,TEMP3
7137 036552                                     LET @TEMP3 :B= DSTMP3
      036552 116777 142001 141724                       MOV     DSTMP3,@TEMP3
7138
7139                                     ;*
7140                                     ; SIGN DSBUF2 BY DATA TYPE
7141                                     ;
7142
7143 036560                                     SELECT TEMP1 OF 5
      036560 016746 141714                               MOV     TEMP1,(SP)
      036564 006316                                     ASL     (SP)
      036566 060716                                     ADD     PC,(SP)
      036570 063607                                     ADD     @ (SP)+,PC
      036572                                     $51525:
      036572 000012                                     .WORD  $51533-$51525
      036574 000052                                     .WORD  $51532-$51525
      036576 000116                                     .WORD  $51531-$51525
      036600 000162                                     .WORD  $51530-$51525
      036602 000226                                     .WORD  $51527-$51525
7144 036604                                     CASE 1
      036604
7145 036604                                     LET DESCR2 := DATA02
      036604 016767 141756 142010                       MOV     DATA02,DESCR2
7146 036612                                     IF SIGN1 EQ #0 THEN
      036612 005767 141634
      036616 001004
      036620                                     LET @TEMP3 :B= @TEMP3 SET.BY #60 ;*
7147 036620 152777 000060 141656
7148 036626                                     ELSE
      036626 000403
      036630
      036630                                     LET @TEMP3 :B= @TEMP3 SET.BY #160;
7149 036630 152777 000160 141646
7150 036636                                     ENDIF
      036636
7151 036636                                     LET DSBUF2-1 :B= #0
      036636 105067 025637                               CLR     DSBUF2-1
7152
7153 036642                                     CASE 2
      036642 000512
      036644
7154 036644                                     LET DESCR2 := DATA02 SET.BY #20000 ;TRAILING OVERPUNCH
      036644 016767 141716 141750                       MOV     DATA02,DESCR2

```



SUBN

7155	036652	052767	020000	141742			BIS	#20000,DESCR2
	036660				LET R5 :B= DSTMP3		MOV	DSTMP3,R5
7156	036660	116705		141673				
	036664				IF SIGN1 EQ #-1 THEN			
	036664	026727	141562	177777			CMP	SIGN1,-1
	036672	001002					BNE	\$51536
7157	036674				LET R5 := R5 + #10.			
	036674	062705	000012				ADD	#10.,R5
7158	036700				ENDIF			
	036700							\$51536:
7159	036700				LET @TEMP3 :B= OVPNCH(R5)			
	036700	116577	056330	141576			MOV	OVPNCH(R5),@TEMP3
7160								
7161	036706				CASE 3			
	036706	000470					BR	\$51526
	036710							\$51531:
7162	036710				LET DESCR2 := DATA02 SET.BY #30000	;LEADING OVERPUNCH		
	036710	016767	141652	141704			MOV	DATA02,DESCR2
	036716	052767	030000	141676			BIS	#30000,DESCR2
7163	036724				LET R5 :B= DSTMP2			
	036724	116705		141626			MOV	DSTMP2,R5
7164	036730				IF SIGN1 EQ #-1 THEN			
	036730	026727	141516	177777			CMP	SIGN1,-1
	036736	001002					BNE	\$51537
7165	036740				LET R5 := R5 + #10.			
	036740	062705	000012				ADD	#10.,R5
7166	036744				ENDIF			
	036744							\$51537:
7167	036744				LET DSBUF2 :B= OVPNCH(R5)			
	036744	116567	056330	025530			MOV	OVPNCH(R5),DSBUF2
7168								
7169	036752				CASE 4			
	036752	000446					BR	\$51526
	036754							\$51530:
7170	036754				LET DESCR2 := DATA02 SET.BY #40000	;TRAILING SEPARATE		
	036754	016767	141606	141640			MOV	DATA02,DESCR2
	036762	052767	040000	141632			BIS	#40000,DESCR2
7171	036770				LET R5 := TEMP3 + #1			
	036770	016705		141510			MOV	TEMP3,R5
	036774	005205					INC	R5
7172	036776				IF SIGN1 EQ #0 THEN			
	036776	005767		141450			TST	SIGN1
	037002	001003					BNE	\$51540
7173	037004				LET (R5) :B= #53			
	037004	112715	000053				MOV	#53,(R5)
7174	037010				ELSE			
	037010	000402					BR	\$51541
	037012							\$51540:
7175	037012				LET (R5) :B= #55			
	037012	112715	000055				MOV	#55,(R5)
7176	037016				ENDIF			
	037016							\$51541:
7177								
7178	037016				CASE 5			
	037016	000424					BR	\$51526
	037020							\$51527:
7179	037020				LET DESCR2 := DATA02 SET.BY #50000	;LEADING SEPARATE		

SUBN

7180	037020	016767	141542	141574			MOV	DATA02,DESCR2
	037026	052767	050000	141566			BIS	#50000,DESCR2
	037034				LET R5 := TEMP3 + #1			
	037034	016705	141444				MOV	TEMP3,R5
	037040	005205					INC	R5
7181	037042				LET (R5) :B= #0			
	037042	105015					CLRB	(R5)
7182	037044				IF SIGN1 EQ #0 THEN			
	037044	005767	141402				TST	SIGN1
	037050	001004					BNE	\$51542
7183	037052				LET DSBUF2-1 :B= #53			
	037052	112767	000053	025421			MOVB	#53,DSBUF2-1
7184	037060				ELSE			
	037060	000403					BR	\$51543
	037062						\$51542:	
7185	037062				LET DSBUF2-1 :B= #55			
	037062	112767	000055	025411			MOVB	#55,DSBUF2 1
7186	037070				ENDIF			
	037070						\$51543:	
7187	037070				ENDSELECT			
	037070						\$51526:	
7188								
7189					:INCR DATTYP FROM #1 TO #5 BY #1 ;DS.EXP LOOP			
7190	037070				INLINE <CLR @#DATTYP>			
	037070	005037	000464				CLR	@#DATTYP
7191	037074				INLINE <INC23: INC DATTYP>			
	037074	005267	141364				INC23:	INC DATTYP
7192	037100				IF DATTYP GT #5 THEN			
	037100	026727	141360	000005			CMP	DATTYP,#5
	037106	003402					BLE	\$51544
7193	037110				INLINE <JMP END23>			
	037110	000167	000642				JMP	END23
7194	037114				ENDIF			
	037114						\$51544:	
7195								
7196					:+			
7197					;	RESTORE MSD & LSD OF DS.EXP		
7198					;-			
7199								
7200	037114				LET DS.EXP :B= DSTMP4 ;MSD			
	037114	116767	141440	025464			MOVB	DSTMP4,DS.EXP
7201	037122				LET TEMP4 := #DS.EXP-1 + DATA04			
	037122	012767	064605	141356			MOV	#DS.EXP-1,TEMP4
	037130	066767	141436	141350			ADD	DATA04,TEMP4
7202	037136				LET @TEMP4 :B= DSTMP5 ;LSD			
	037136	116777	141417	141342			MOVB	DSTMP5,@TEMP4
7203								
7204					:+			
7205					;	SIGN DS.EXP BY DATA TYPE		
7206					;			
7207					;			
7208	037144				SELECT DATTYP OF 5			
	037144	016746	141314				MOV	DATTYP, (SP)
	037150	006316					ASL	(SP)
	037152	060716					ADD	PC,(SP)
	037154	063607					ADD	@(SP),PC
	037156						\$51545:	

SUBN

037156	000012						.WORD	\$51553	\$51545
037160	000060						.WORD	\$51552-	\$51545
037162	000136						.WORD	\$51551-	\$51545
037164	000214						.WORD	\$51550-	\$51545
037166	000260						.WORD	\$51547-	\$51545
7209	037170								
	037170							\$51553:	
7210	037170	016767	141370	141430			LET DESCR4 := DATA04		;SIGNED ZONED
	037176						IF DATA04 NE #0 THEN		
7211	037176	005767	141370						
	037202	001414							
7212	037204						IF SIGN2 EQ #0 THEN		
	037204	005767	141244						
	037210	001004							
7213	037212						LET @TEMP4 :B= @TEMP4 SET.BY #60		
	037212	152777	000060	141266					
7214	037220						ELSE		
	037220	000403							
	037222								
7215	037222						LET @TEMP4 :B= @TEMP4 SET.BY #160		
	037222	152777	000160	141256					
7216	037230						ENDIF		
	037230								
7217	037230						LET DS.EXP-1 :B= #0		
	037230	105067	025351						
7218	037234						ENDIF		
	037234								
7219									
7220	037234								
	037234	000524							
	037236								
7221	037236						LET DESCR4 := DATA04 SET.BY #20000		;TRAILING OVERPUNCH
	037236	016767	141330	141362					
	037244	052767	020000	141354					
7222	037252						IF DATA04 NE #0 THEN		
	037252	005767	141314						
	037256	001415							
7223	037260						LET R5 :B= DSTMP5		
	037260	116705	141275						
7224	037264						LET R5 := R5 - #60		
	037264	162705	000060						
7225	037270						IF SIGN2 EQ #-1 THEN		
	037270	026727	141160	177777					
	037276	001002							
7226	037300						LET R5 :- R5 + #10.		
	037300	062705	000012						
7227	037304						ENDIF		
	037304								
7228	037304						LET @TEMP4 :B= OVPNCH(R5		
	037304	116577	056330	141174					
7229	037312						ENDIF		
	037312								
7230									
7231	037312								
	037312	000475							
	037314								

SUBN							
7232	037314				LET DESCR4 := DATA04 SET.BY #30000	;LEADING OVERPUNCH	
	037314	016767	141252	141304			MOV DATA04,DESCR4
	037322	052767	030000	141276			BIS #30000,DESCR4
7233	037330				IF DATA04 NE #0 THEN		
	037330	005767	141236				TST DATA04
	037334	001415					BEQ \$51561
7234	037336				LET R5 :B= DSTMP4		
	037336	116705	141216				MOVB DSTMP4,R5
7235	037342				LET R5 := R5 #60		
	037342	162705	000060				SUB #60,R5
7236	037346				IF SIGN2 EQ # 1 THEN		
	037346	026727	141102	177777			CMP SIGN2,#-1
	037354	001002					BNE \$51562
7237	037356				LET R5 := R5 + #10.		
	037356	062705	000012				ADD #10.,R5
7238	037362				ENDIF		
	037362						\$51562:
7239	037362				LET DS.EXP :B= OVPNCH(R5)		
	037362	116567	056330	025216			MOVB OVPNCH(R5),DS.EXP
7240	037370				ENDIF		
	037370						\$51561:
7241							
7242	037370				CASE 4		
	037370	000446					BR \$51546
	037372						\$51550:
7243	037372				LET DESCR4 := DATA04 SET.BY #40000	;TRAILING SEPARATE	
	037372	016767	141174	141226			MOV DATA04,DESCR4
	037400	052767	040000	141220			BIS #40000,DESCR4
7244	037406				LET R5 := TEMP4 + #1		
	037406	016705	141074				MOV TEMP4,R5
	037412	005205					INC R5
7245	037414				IF SIGN2 EQ #0 THEN		
	037414	005767	141034				TST SIGN2
	037420	001003					BNE \$51563
7246	037422				LET (R5) :B= #53		
	037422	112715	000053				MOVB #53,(R5)
7247	037426				ELSE		
	037426	000402					BR \$51564
	037430						\$51563:
7248	037430				LET (R5) :B= #55		
	037430	112715	000055				MOVB #55,(R5)
7249	037434				ENDIF		
	037434						\$51564:
7250							
7251	037434				CASE 5		
	037434	000424					BR \$51546
	037436						\$51547:
7252	037436				LET DESCR4 := DATA04 SET.BY #50000		
	037436	016767	141130	141162			MOV DATA04,DESCR4
	037444	052767	050000	141154			BIS #50000,DESCR4
7253	037452				LET R5 := TEMP4 + #1		
	037452	016705	141030				MOV TEMP4,R5
	037456	005205					INC R5
7254	037460				LET (R5) :B= #0		
	037460	105015					CLRB (R5)
7255	037462				IF SIGN2 EQ #0 THEN		
	037462	005767	140766				TST SIGN2

SUBN

7256	037466	001004			LET DS.EXP-1 :B= #53	BNE	\$51565
	037470					MOVB	#53,DS.EXP 1
7257	037470	112767	000053	025107	ELSE		
	037476					BR	\$51566
	037500					\$51565:	
7258	037500				LET DS.EXP-1 :B= #55		
	037500	112767	000055	025077		MOVB	#55,DS.EXP-1
7259	037506				ENDIF	\$51566:	
	037506					\$51546:	
7260	037506				ENDSELECT		
	037506						
7261							
7262					:+		
7263					; CLR DS.DST & LOAD DESCRIPTORS		
7264					;-		
7265							
7266	037506				LET R0 := #DS.DST 1		
	037506	012700	064543			MOV	#DS.DST-1,R0
7267	037512				LET R1 := #41		
	037512	012701	000041			MOV	#41,R1
7268	037516				INLINE <DOIT: CLR (R0)>		
	037516	105020				DOIT:	CLR (R0).
7269	037520				INLINE <SOB R1, DOIT>		
	037520	077102				SOB	R1, DOIT
7270	037522				LET R0 := DESCRO		
	037522	016700	141070			MOV	DESCRO,R0
7271	037526				LET R1 := DESCR1		
	037526	016701	141066			MOV	DESCR1,R1
7272	037532				LET R2 := DESCR2		
	037532	016702	141064			MOV	DESCR2,R2
7273	037536				LET R3 := DESCR3		
	037536	016703	141062			MOV	DESCR3,R3
7274	037542				LET R4 := DESCR4		
	037542	016704	141060			MOV	DESCR4,R4
7275	037546				LET R5 := DESCR5		
	037546	016705	141056			MOV	DESCR5,R5
7276	037552				LET R4.EXP := R4		
	037552	010467	140756			MOV	R4,R4.EXP
7277	037556				LET R5.EXP := R5		
	037556	010567	140754			MOV	R5,R5.EXP
7278	037562				INLINE <SUBN>		
	037562	076051				SUBN	
7279							
7280					:+		
7281					; CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.		
7282					;-		
7283							
7284	037564				LET CC := PSW		
	037564	016767	140206	140726		MOV	PSW,CC
7285	037572				LET CC := CC CLR.BY #177760		
	037572	042767	177760	140720		BIC	#177760,CC
7286	037600				IF CC NE CC.EXP THEN		
	037600	026767	140714	140714		CMP	CC,CC.EXP
	037606	001402				BEQ	\$51567
7287	037610				CALL ERROR		
	037610	004767	020064			JSR	PC,ERROR

```

SUBN
7288 037614          ENDIF
      037614
7289 037614          IF R0 NE R0.EXP THEN
      037614 020067 140704
      037620 001402
7290 037622          CALL ERROR
      037622 004767 020052
7291 037626          ENDIF
      037626
7292 037626          IF R1 NE R1.EXP THEN
      037626 020167 140674
      037632 001402
7293 037634          CALL ERROR
      037634 004767 020040
7294 037640          ENDIF
      037640
7295 037640          IF R2 NE R2.EXP THEN
      037640 020267 140664
      037644 001402
7296 037646          CALL ERROR
      037646 004767 020026
7297 037652          ENDIF
      037652
7298 037652          IF R3 NE R3.EXP THEN
      037652 020367 140654
      037656 001402
7299 037660          CALL ERROR
      037660 004767 020014
7300 037664          ENDIF
      037664
7301 037664          IF R4 NE R4.EXP THEN
      037664 020467 140644
      037670 001402
7302 037672          CALL ERROR
      037672 004767 020002
7303 037676          ENDIF
      037676
7304 037676          IF R5 NE R5.EXP THEN
      037676 020567 140634
      037702 001402
7305 037704          CALL ERROR
      037704 004767 017770
7306 037710          ENDIF
      037710
7307
7308
7309
7310
7311
7312
7313 037710          LET R0 := #DS.DST 1
      037710 012700 064543
7314 037714          LET R1 := #DS.EXP-1
      037714 012701 064605
7315 037720          LET ERRFLG := #0
      037720 005067 140532
7316 037724          WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO

```

\$51567:

```

CMP R0,R0.EXP
BEQ $51570

```

```

JSR PC,ERROR

```

\$51570:

```

CMP R1,R1.EXP
BEQ $51571

```

```

JSR PC,ERROR

```

\$51571:

```

CMP R2,R2.EXP
BEQ $51572

```

```

JSR PC,ERROR

```

\$51572:

```

CMP R3,R3.EXP
BEQ $51573

```

```

JSR PC,ERROR

```

\$51573:

```

CMP R4,R4.EXP
BEQ $51574

```

```

JSR PC,ERROR

```

\$51574:

```

CMP R5,R5.EXP
BEQ $51575

```

```

JSR PC,ERROR

```

\$51575:

```

;+
; CHECK RESULTS VS EXPECTED RESULTS
;

```

```

MOV #DS.DST-1,R0

```

```

MOV #DS.EXP-1,R1

```

```

CLR ERRFLG

```

SUBN

037724						\$51576:	
037724	020027	064604				CMP	RO, #DS.DST.40
037730	001410					BEQ	\$51577
037732	005767	140520				TST	ERRFLG
037736	001005					BNE	\$51577
7317	037740			IFB (R0)+ NE (R1)+ THEN			
	037740	122021				CMPB	(R0)+,(R1)+
	037742	001402				BEQ	\$51600
7318	037744			CALL ERROR		JSR	PC.ERROR
7319	037750			ENDIF			
	037750					\$51600:	
7320	037750			ENDDO			
	037750	000765				BR	\$51576
	037752					\$51577:	
7321				;ENDINC			
7322	037752			INLINE <JMP INC23>			JMP INC23
	037752	000167	177116				
7323	037756			INLINE <END23:>			END23:
	037756						
7324				;ENDINC			
7325	037756			INLINE <JMP INC22>			JMP INC22
	037756	000167	176522				
7326	037762			INLINE <END22:>			END22:
	037762						
7327				;ENDINC			
7328	037762			INLINE <JMP INC21>			JMP INC21
	037762	000167	176132				
7329	037766			INLINE <END21:>			END21:
	037766						
7330	037766			LET TSTCAS := TSTCAS + #22			ADD #22,TSTCAS
	037766	062767	000022	140550			
7331	037774			LET COUNT := COUNT #1			DEC COUNT
	037774	005367	140470				
7332				;UNTIL COUNT EQ #0			
7333	040000			IF COUNT NE #0 THEN			TST COUNT
	040000	005767	140464				BEQ \$51601
	040004	001402					
7334	040006			INLINE <JMP \$5051>			JMP \$5051
	040006	000167	175530				
7335	040012			ENDIF			
	040012					\$51601:	
7336							
7337				;+			
7338				;	INLINE CASE - SUBNI, USING UNSIGNED ZONED DATA		
7339				; -			
7340							
7341	040012			CALL CLRDSB			JSR PC.CLRDSB
	040012	004767	016656				
7342							
7343				;+			
7344				;	LOAD DSBUF1		
7345				; -			
7346							
7347	040016			LET R0 := #ADD1+2	;	SOURCE	MOV #ADD1.2,R0
	040016	012700	065430				
7348	040022			LET R1 := TADDI CLR.BY #10000	;	# OF BYTES TO XFER	

SUBN

040022	016701	026074				MOV	TADDI,R1
040026	042701	010000				BIC	#10000,R1
7349 040032			LET R2 := #DSBUF1		;DESTINATION		
040032	012702	064440				MOV	#DSBUF1,R2
7350 040036			INLINE <1\$:	MOVB	(R0)+, (R2)+>	1\$:	MOVB (R0)+, (R2)
040036	112022						
7351 040040			INLINE <	SOB	R1, 1\$>		
040040	077102					SOB	R1, 1\$
7352							
7353			;+				
7354			LOAD DSBUF2				
7355			;				
7356			;-				
7357 040042			LET R0 := #ADD1A+2				
040042	012700	065452				MOV	#ADD1A+2,R0
7358 040046			LET R1 := TADDI+4 CLR.BY #10000				
040046	016701	026054				MOV	TADDI+4,R1
040052	042701	010000				BIC	#10000,R1
7359 040056			LET R2 := #DSBUF2				
040056	012702	064502				MOV	#DSBUF2,R2
7360 040062			INLINE <2\$:	MOVB	(R0)+, (R2)+>	2\$:	MOVB (R0)+, (R2)
040062	112022						
7361 040064			INLINE <	SOB	R1, 2\$>		
040064	077102					SOB	R1, 2\$
7362							
7363			;+				
7364			LOAD DS.EXP				
7365			;				
7366			;				
7367 040066			LET R0 := #ADD1C+2				
040066	012700	065524				MOV	#ADD1C+2,R0
7368 040072			LET R1 := TADDI+10 CLR.BY #10000				
040072	016701	026034				MOV	TADDI+10,R1
040076	042701	010000				BIC	#10000,R1
7369 040102			LET R2 := #DS.EXP				
040102	012702	064606				MOV	#DS.EXP,R2
7370 040106			INLINE <3\$:	MOVB	(R0)+, (R2)+>	3\$:	MOVB (R0)+, (R2)
040106	112012						
7371 040110			INLINE <	BISB	#60, (R2)+>		
040110	152722	000060				BISB	#60, (R2)
7372 040114			INLINE <	SOB	R1, 3\$>		
040114	077104					SOB	R1, 3\$
7373							
7374 040116			CALL CLRREG		;ENSURE R0-R5 ARE CLR		
040116	004767	016316				JSR	PC,CLRREG
7375							
7376 040122			INLINE <SUBNI>				
040122	076151					SUBNI	
7377 040124			INLINE <.WORD TADDI>				
040124	066122					.WORD TADDI	
7378 040126			INLINE <.WORD TADDI+4>				
040126	066126					.WORD TADDI+4	
7379 040130			INLINE <.WORD TADDI+10>				
040130	066132					.WORD TADDI+10	
7380 040132			LET CC := PSW				
040132	016767	137640	140360			MOV	PSW,CC
7381 040140			LET CC := CC CLR.BY #177760				



SUBN

```

7382 040140 042767 177760 140352          BIC      #177760,CC
      040146          IF CC NE #0 THEN          TST      CC
      040146 005767 140346          BEQ      $51602
7383 040152 001402          CALL ERROR          JSR      PC,ERROR
      040154 004767 017520          ENDIF
7384 040160          $51602:
      040160          ;+
7385          ;          ENSURE GPR'S STILL ZERO
7386          ;
7387          ;-
7388
7389          IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN
7390 040160          TST      R0
      040160 005700          BNE      $51603
      040162 001004          TST      R1
      040164 005701          BNE      $51603
      040166 001002          TST      R2
      040170 005702          BEQ      $51604
      040172 001402          $51603:
      040174          CALL ERROR          JSR      PC,ERROR
7391 040174 004767 017500          ENDIF
7392 040200          $51604:
      040200          IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
7393 040200          TST      R3
      040200 005703          BNE      $51605
      040202 001004          TST      R4
      040204 005704          BNE      $51605
      040206 001002          TST      R5
      040210 005705          BEQ      $51606
      040212 001402          $51605:
      040214          CALL ERROR          JSR      PC,ERROR
7394 040214 004767 017460          ENDIF
7395 040220          $51606:
      040220          ;+
7396          ;          CHECK RESULTS
7397          ;
7398          ;-
7399
7400          LET R0 := #DS.DST
7401 040220 012700 064544          MOV      #DS.DST,R0
      040220          LET R1 := #DS.EXP          ;EXPECTED DATA          MOV      #DS.EXP,R1
7402 040224 012701 064606          LET ERRFLG := #0          CLR      ERRFLG
7403 040230 005067 140222          WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO
7404 040234          $51607:
      040234 020027 064604          CMP      R0,#DS.DST+40
      040240 001410          BEQ      $51610
      040242 005767 140210          TST      ERRFLG
      040246 001005          BNE      $51610
7405 040250          IFB (R0)+ NE (R1)+ THEN          CMPB    (R0), (R1)+
      040250 122021          BEQ      $51611
      040252 001402

```

SUBN

```
7406 040254          CALL ERROR
      040254 004767 017420
7407 040260          ENDIF
      040260
7408 040260          ENDDO
      040260 000765
      040262
7409
7410
```

JSR PC.ERROR  
\$51611:  
BR \$51607  
\$51610:

CMPN

```

7412      .SBTTL  CMPN
7413      ;*****
7414      ;
7415      ;   CMPN TEST
7416      ;
7417      ;*****
7418
7419 040262 ROUTINE XCMPN
7420 040262                                XCMPN:
7421 040262 000004                          SCOPE
7422 040264 000240                          NOP
7423 040266 012767 000020 140110          ;DO 20 ITERATIONS
7424 040274 016767 024346 140166          ;GET TEST COUNT
7425 040302 012767 064650 140234          ;POINT TO FIRST TEST CASE
7426 040310                                ;REPEAT
7427 040310 004767 016360                  ;REPEAT
7428 040314 005067 140202                  ;REPEAT
7429 040320 005067 140124                  ;REPEAT
7430 040324 005067 140134                  ;REPEAT
7431
7432      ;+
7433      ; GET SOURCE DESCRIPTORS FROM TABLE
7434      ;-
7435
7436 040330 LET R0 := TSTCAS
7437 040334 016700 140210                  ;SRC1 LENGTH
7438 040340 012067 140222                  ;SRC1 POINTER
7439 040344 012067 140220                  ;SRC2 LENGTH
7440 040350 011067 140214                  ;SRC2 POINTER
7441
7442      ;+
7443      ; LOAD SRC1 AND SRC2 INTO DSBUF1 AND DSBUF2.
7444      ;-
7445
7446 040354 LET R2 := DATA01 + #2          ;GET SRC1 ADDRESS
7447 040360 016702 140204                  ;GET SRC1 ADDRESS
7448 040364 012703 064440                  ;GET SRC1 ADDRESS
7449 040370 111267 140160                  ;GET SRC1 ADDRESS

```

CMPN

```

7449 040374          DECR R1 FROM DATA00 TO #1 BY #1
      040374 016701 140162
      040400 000401
      040402
      040402 005301
      040404
      040404 020127 000001
      040410 002402
7450 040412          LET (R3)+ :B= (R2)+
      040412 112223
7451 040414          ENDDEC
      040414 000772
      040416
7452 040416          IF DATA00 HI #0 THEN
      040416 005767 140140
      040422 101402
      040424          LET DSTMP1 :B= -(R2)          ;SAVE SRC1 LSD
7453 040424          ENDIF
      040424 114267 140125
7454 040430          LET R2 := DATA03 + #2          ;GET SRC2 ADDRESS
      040430
7455 040430          LET R3 := #DSBUF2
      040430 016702 140134
      040434 062702 000002
7456 040440          LET DSTMP2 :B= (R2)          ;SAVE SRC2 MSD
      040440 012703 064502
7457 040444          DECR R1 FROM DATA02 TO #1 BY #1
      040444 111267 140106
7458 040450          LET (R3)+ :B= (R2)+
      040450 016701 140112
      040454 000401
      040456 005301
      040460
      040460 020127 000001
      040464 002402
7459 040466          ENDDEC
      040466 112223
7460 040470          LET DSTMP3 :B= -(R2)          ;SAVE SRC2 LSD
      040470 000772
      040472
7461 040472          ;+
      040472 114267 140061          ; COMPARE SRC1 AND SRC2, AND SET N AND Z BITS
7462
7463          ; IN CC.EXP ACCORDINGLY.
7464          ;-
7465
7466
7467
7468 040476          IF DATA00 EQ #0 AND DATA02 EQ #0 THEN          ;SRC1 LENGTH AND SRC2 LENGTH = 0
      040476 005767 140060
      040502 001007
      040504 005767 140056
      040510 001004
7469 040512          LET CC.EXP := CC.EXP SET.BY #ZBIT
      040512 052767 000004 140002
7470 040520          ELSE
      040520 000424

```





CMPN

```

7513 041030          ELSE                               ;SOURCES WERE EQUAL
      041030 000403                                     BR      $51646
      041032                                     $51642:
7514 041032          LET CC.EXP := CC.EXP SET.BY #ZBIT
      041032 052767 000004 137462                     BIS      #ZBIT,CC.EXP
7515 041040          ENDIF                               $51646:
      041040                                     $51637:
7516 041040          ENDIF                               $51633:
      041040                                     $51630:
7517 041040          ENDIF
      041040
7518 041040          ENDIF
      041040
7519
7520
7521 ;*
7522 ; PERFORM THE CMPN INSTRUCTION WITH ALL COMBINATIONS
7523 ; OF SIGNED ZONE, TRAILING AND LEADING OVERPUNCH, AND
7524 ; TRAILING AND LEADING SEPARATE (UNSIGNED ZONE WILL
7525 ; BE TESTED IN THE INLINE PORTION OF THE TEST).
7526 ; CHECK THE RESULTS AND REPORT ERRORS. "DATTYP"
7527 ; WILL CONTAIN SRC1 DATA TYPE AND "TEMP1" WILL
7528 ; CONTAIN SRC2 DATA TYPE CURRENTLY BEING TESTED.
7529 ;
7530 ;REPEAT                               ;SRC1 LOOP
7531 041040          INLINE <$5201:>
      041040
7532 041040          IF DATTYP EQ #10000 THEN           ;SKIP UNSIGNED ZONED
      041040 026727 137420 010000                       CMP      DATTYP,#10000
      041046 001003                                       BNE      $51647
7533 041050          LET DATTYP := DATTYP + #10000
      041050 062767 010000 137406                       ADD      #10000,DATTYP
7534 041056          ENDIF
      041056
7535 ;RESTORE THE MSD AND LSD OF SRC1
7536 041056          LET DSBUF1 :B= DSTMPO               ;SRC1 MSD
      041056 116767 137472 023354                       MOVB     DSTMPO,DSBUF1
7537 041064          LET RO := #DSBUF1-1 + DATA00
      041064 012700 064437                               MOV      #DSBUF1-1,RO
      041070 066700 137466                               ADD      DATA00,RO
7538 041074          LET (RO) :B= DSTMP1               ;SRC1 LSD
      041074 116710 137455                               MOVB     DSTMP1,(RO)
7539 ;PUT THE PROPER SIGN INTO SRC1, ACCORDING TO DATTYP.
7540 041100          IF DATTYP EQ #0 THEN               ;*** SIGNED ZONED ***
      041100 005767 137360                               TST      DATTYP
      041104 001017                                       BNE      $51650
7541 041106          LET RO := #DSBUF1-1 + DATA00
      041106 012700 064437                               MOV      #DSBUF1-1,RO
      041112 066700 137444                               ADD      DATA00,RO
7542 041116          IF @DATA01 EQ #0 THEN             ;SIGN IS +
      041116 005777 137442                               TST      @DATA01
      041122 001003                                       BNE      $51651
7543 041124          LET (RO) :B= (RO) SET.BY #60
      041124 152710 000060                               BISB     #60,(RO)
7544 041130          ELSE                               ;SIGN IS -
      041130 000402                                     BR      $51652
      041132                                     $51651:

```

```

CMPN
7545 041132          LET (R0) :B= (R0) SET.BY #160
      041132 152710 000160
7546 041136          ENDIF
      041136
7547 041136          LET R0 := #DSBUF1 1
      041136 012700 064437
7548 041142          LET (R0) :B= #0          ;RESTORE AFTER LEADING SEPARATE
      041142 105010
7549 041144          ENDIF
      041144
7550 041144          IF DATTYP EQ #20000 THEN          ;*** TRAILING OVERPUNCH ***
      041144 026727 137314 020000
      041152 001021
7551 041154          IF DATA00 HI #0 THEN
      041154 005767 137402
      041160 101416
7552 041162          LET R0 := #DSBUF1-1 + DATA00
      041162 012700 064437
      041166 066700 137370
7553 041172          IF @DATA01 EQ #0 THEN          ;SIGN IS +
      041172 005777 137366
      041176 001002
7554 041200          LET R1 := #0
      041200 005001
7555 041202          ELSE
      041202 000402
      041204
7556 041204          LET R1 := #10.
      041204 012701 000012
7557 041210          ENDIF
      041210
7558 041210          LET R1 :B= R1 SET.BY (R0)          ;FORM TABLE OFFSET
      041210 151001
7559 041212          LET (R0) :B= OVPNCH(R1)          ;PUT IN SIGN/DIGIT
      041212 116110 056330
7560 041216          ENDIF
      041216
7561 041216          ENDIF
      041216
7562 041216          IF DATTYP EQ #30000 THEN          ;*** LEADING OVERPUNCH ***
      041216 026727 137242 030000
      041224 001017
7563 041226          IF DATA00 HI #0 THEN
      041226 005767 137330
      041232 101414
7564 041234          LET R0 := #DSBUF1
      041234 012700 064440
7565 041240          IF @DATA01 EQ #0 THEN          ;SIGN IS -
      041240 005777 137320
      041244 001002
7566 041246          LET R1 := #0
      041246 005001
7567 041250          ELSE
      041250 000402
      041252
7568 041252          LET R1 := #10.
      041252 012701 000012

```



CMPN

```

7569 041256          ENDIF
      041256          $51662:
7570 041256          LET R1 :B= R1 SET.BY (R0) ;FORM TABLE OFFSET
      041256 151001          ;
7571 041260          LET (R0) :B= OVPNCH(R1) ;PUT IN SIGN/DIGIT      BISB   (R0),R1
      041260 116110 056330          ;
7572 041264          ENDIF
      041264          $51660:
7573 041264          ENDIF
      041264          $51657:
7574 041264          IF DATTYP EQ #40000 THEN ;*** TRAILING SEPARATE ***
      041264 026727 137174 040000          CMP     DATTYP,#40000
      041272 001014          BNE     $51663
7575 041274          LET R0 := #DSBUF1 + DATA00
      041274 012700 064440          MOV     #DSBUF1,R0
      041300 066700 137256          ADD     DATA00,R0
7576 041304          IF @DATA01 EQ #0 THEN ;SIGN IS +
      041304 005777 137254          TST     @DATA01
      041310 001003          BNE     $51664
7577 041312          LEI (R0) :B- #53
      041312 112710 000053          MOVB   #53,(R0)
7578 041316          ELSE ;SIGN IS -
      041316 000402          BR     $51665
      041320          $51664:
7579 041320          LET (R0) :B= #55
      041320 112710 000055          MOVB   #55,(R0)
7580 041324          ENDIF
      041324          $51665:
7581 041324          ENDIF
      041324          $51663:
7582 041324          IF DATTYP EQ #50000 THEN ;*** LEADING SEPARATE ***
      041324 026727 137134 050000          CMP     DATTYP,#50000
      041332 001017          BNE     $51666
7583 041334          LET R0 := #DSBUF1-1
      041334 012700 064437          MOV     #DSBUF1-1,R0
7584 041340          IF @DATA01 EQ #0 THEN ;SIGN IS +
      041340 005777 137220          TST     @DATA01
      041344 001003          BNE     $51667
7585 041346          LET (R0) :B= #53
      041346 112710 000053          MOVB   #53,(R0)
7586 041352          ELSE ;SIGN IS
      041352 000402          BR     $51670
      041354          $51667:
7587 041354          LET (R0) :B= #55
      041354 112710 000055          MOVB   #55,(R0)
7588 041360          ENDIF
      041360          $51670:
7589 041360          LET R0 := #DSBUF1 + DATA00
      041360 012700 064440          MOV     #DSBUF1,R0
      041364 066700 137172          ADD     DATA00,R0
7590 041370          LET (R0) :B= #0 ;RESTORE AFTER TRAILING SEPARATE
      041370 105010          CLR     (R0)
7591 041372          ENDIF
      041372          $51666:
7592 041372          LET TEMP1 :- #0 ;CLEAR SRC2 DATA TYPE
      041372 005067 137102          CLR     TEMP1
7593          ;REPEAT ;SRC2 LOOP

```

CMPN

7594	041376				INLINE <\$5202:>			
	041376							\$5202:
7595	041376				IF TEMP1 EQ #10000 THEN			;SKIP UNSIGNED ZONED
	041376	026727	137076	010000				CMP TEMP1,#10000
	041404	001003						BNE \$51671
7596	041406				LET TEMP1 := TEMP1 + #10000			ADD #10000,TEMP1
	041406	062767	010000	137064				
7597	041414				ENDIF			\$51671:
	041414							
7598					;RESTORE THE MSD AND LSD OF SRC2			
7599	041414				LET DSBUF2 :B= DSTMP2			;SRC2 MSD
	041414	116767	137136	023060				MOVB DSTMP2,DSBUF2
7600	041422				LET R0 := #DSBUF2-1 + DATA02			MOV #DSBUF2-1,R0
	041422	012700	064501					ADD DATA02,R0
	041426	066700	137134					
7601	041432				LET (R0) :B= DSTMP3			;SRC2 LSD
	041432	116710	137121					MOVB DSTMP3,(R0)
7602					;PUT THE PROPER SIGN INTO SRC2, ACCORDING TO TEMP1			
7603	041436				IF TEMP1 EQ #0 THEN			;*** SIGNED ZONED ***
	041436	005767	137036					TST TEMP1
	041442	001017						BNE \$51672
7604	041444				LET R0 := #DSBUF2-1 + DATA02			MOV #DSBUF2-1,R0
	041444	012700	064501					ADD DATA02,R0
	041450	066700	137112					
7605	041454				IF @DATA03 EQ #0 THEN			;SIGN IS +
	041454	005777	137110					TST @DATA03
	041460	001003						BNE \$51673
7606	041462				LET (R0) :B= (R0) SET.BY #60			BISB #60,(R0)
	041462	152710	000060					
7607	041466				ELSE			;SIGN IS -
	041466	000402						BR \$51674
	041470							\$51673:
7608	041470				LET (R0) :B= (R0) SET.BY #160			BISB #160,(R0)
	041470	152710	000160					
7609	041474				ENDIF			\$51674:
	041474							
7610	041474				LET R0 := #DSBUF2-1			MOV #DSBUF2-1,R0
	041474	012700	064501					ADD DATA02,R0
7611	041500				LET (R0) :B= #0			;RESTORE AFTER LEADING SEPARATE
	041500	105010						CLRB (R0)
7612	041502				ENDIF			\$51672:
	041502							;*** TRAILING OVERPUNCH ***
7613	041502				IF TEMP1 EQ #20000 THEN			CMP TEMP1,#20000
	041502	026727	136772	020000				BNE \$51675
	041510	001016						
7614	041512				LET R0 := #DSBUF2-1 + DATA02			MOV #DSBUF2-1,R0
	041512	012700	064501					ADD DATA02,R0
	041516	066700	137044					
7615	041522				IF @DATA03 EQ #0 THEN			;SIGN IS +
	041522	005777	137042					TST @DATA03
	041526	001002						BNE \$51676
7616	041530				LET R1 := #0			CLR R1
	041530	005001						
7617	041532				ELSE			;SIGN IS -
	041532	000402						BR \$51677
	041534							\$51676:
7618	041534				LET R1 := #10.			

CMPN

7619	041534	012701	000012				MOV	#10.,R1
	041540			ENDIF				
7620	041540			LET R1 :B= R1 SET.BY (R0)		\$51677:		
	041540	151001					;FORM TABLE OFFSET	
7621	041542			LET (R0) :B= OVPNCH(R1)			BISB	(R0),R1
	041542	116110	056330				;PUT IN SIGN/DIGIT	
7622	041546			ENDIF			MOVB	OVPNCH(R1),(R0)
	041546							
7623	041546			IF TEMP1 EQ #30000 THEN		\$51675:		
	041546	026727	136726				;*** LEADING OVERPUNCH ***	
	041554	001014	030000				CMP	TEMP1,#30000
7624	041556			LET R0 := #DSBUF2			BNE	\$51700
	041556	012700	064502				MOV	#DSBUF2,R0
7625	041562			IF @DATA03 EQ #0 THEN				
	041562	005777	137002				;SIGN IS +	
	041566	001002					TST	@DATA03
7626	041570			LET R1 := #0			BNE	\$51701
	041570	005001					CLR	P1
7627	041572			ELSE				
	041572	000402					;SIGN IS	
	041574						BR	\$51702
7628	041574			LET R1 := #10.		\$51701:		
	041574	012701	000012				MOV	#10.,R1
7629	041600			ENDIF				
	041600							
7630	041600			LET R1 :B= R1 SET.BY (R0)		\$51702:		
	041600	151001					;FORM TABLE OFFSET	
7631	041602			LET (R0) :B= OVPNCH(R1)			BISB	(R0),R1
	041602	116110	056330				;PUT IN SIGN/DIGIT	
7632	041606			ENDIF			MOVB	OVPNCH(R1),(R0)
	041606							
7633	041606			IF TEMP1 FQ #40000 THEN		\$51700:		
	041606	026727	136666				;*** TRAILING SEPARATE ***	
	041614	001014	040000				CMP	TEMP1,#40000
7634	041616			LET R0 := #DSBUF2 + DATA02			BNE	\$51703
	041616	012700	064502				MOV	#DSBUF2,R0
	041622	066700	136740				ADD	DATA02,R0
7635	041626			IF @DATA03 EQ #0 THEN				
	041626	005777	136736				;SIGN IS +	
	041632	001003					TST	@DATA03
7636	041634			LET (R0) :B= #53			BNE	\$51704
	041634	112710	000053				MOVB	#53,(R0)
7637	041640			ELSE				
	041640	000402					;SIGN IS -	
	041642						BR	\$51705
7638	041642			LET (R0) :B= #55		\$51704:		
	041642	112710	000055				MOVB	#55,(R0)
7639	041646			ENDIF				
	041646							
7640	041646			ENDIF				
	041646							
7641	041646			IF TEMP1 EQ #50000 THEN		\$51703:		
	041646	026727	136626				;*** LEADING SEPARATE ***	
	041654	001017	050000				CMP	TEMP1,#50000
7642	041656			LET R0 := #DSBUF2-1			BNE	\$51706
	041656	012700	064501				MOV	#DSBUF2-1,R0

CMPN

```

7643 041662          IF @DATA03 EQ #0 THEN          ;SIGN IS *
      041662 005777 136702
      041666 001003
7644 041670          LET (R0) :B= #53
      041670 112710 000053
7645 041674          ELSE          ;SIGN IS
      041674 000402
      041676
7646 041676          LET (R0) :B= #55
      041676 112710 000055
7647 041702          ENDIF
      041702
7648 041702          LET R0 := #DSBUF2 + DATA02
      041702 012700 064502
      041706 066700 136654
7649 041712          LET (R0) :B= #0          ;RESTORE AFTER TRAILING SEPARATE
      041712 105010
7650 041714          ENDIF
      041714
7651
7652
7653          ;+
7654          ; LOAD THE GPR'S AND DO THE CMPN INSTRUCTION
7655          ; -
7656 041714          LET R0 :- DATTYP SET.BY DATA00          ;SRC1.DSCR
      041714 016700 136544
      041720 056700 136636
7657 041724          LET R1 := #DSBUF1          ; "
      041724 012701 064440
7658 041730          LET R2 := TEMP1 SET.BY DATA02          ;SRC2.DSCR
      041730 016702 136544
      041734 056702 136626
7659 041740          LET R3 := #DSBUF2          ; "
      041740 012703 064502
7660 041744          LET R4 := #0
      041744 005004
7661 041746          LET R5 := #0
      041746 005005
7662 041750          INLINE <CMPN>
      041750 076052
7663
7664
7665          ;+
7666          ; CHECK THE RESULTS
7667          ; -
7668 041752          LET CC := PSW
      041752 016767 136020 136540
7669 041760          LET CC := CC CLR.BY #177760
      041760 042767 177760 136532
7670 041766          IF CC NE CC.EXP THEN
      041766 026767 136526 136526
      041774 001402
7671 041776          CALL ERROR
      041776 004767 015676
7672 042002          ENDIF
      042002
7673 042002          IF R0 NE #0 THEN

```

```

TST @DATA03
BNE $51707
MOVB #55,(R0)
BR $51710
$51707:
MOVB #55,(R0)
$51710:
MOV #DSBUF2,R0
ADD DATA02,R0
CLRB (R0)
$51706:
MOV DATTYP,R0
BIS DATA00,R0
MOV #DSBUF1,R1
MOV TEMP1,R2
BIS DATA02,R2
MOV #DSBUF2,R3
CLR R4
CLR R5
CMP CC,CC.EXP
BEQ $51711
JSR PC.ERROR
$51711:

```



CMPN

```

7699 042122          INLINE <JMP $5201>
      042122 000167 176712
7700 042126          ENDIF
      042126
      $51721:
7701
7702 042126          LET TSTCAS := TSTCAS + #10
      042126 062767 000010 136410
7703 042134          LET COUNT := COUNT - #1
      042134 005367 136330
7704          ;UNTIL COUNT EQ #0 THEN
7705 042140          IF COUNT NE #0 THEN
      042140 005767 136324
      042144 001402
7706 042146          INLINE <JMP $5052>
      042146 000167 176136
7707 042152          ENDIF
      042152
      $51722:
7708
7709
7710          ;*
7711          ; DO ONE INLINE CASE - CMPNI. USE UNSIGNED ZONED DATA TYPE.
7712          ;-
7713
7714 042152          LET R0 := #-1
      042152 012700 177777
7715 042156          LET R1 := #-1
      042156 012701 177777
7716 042162          LET R2 := #-1
      042162 012702 177777
7717 042166          LET R3 := #-1
      042166 012703 177777
7718 042172          LET R4 := #-1
      042172 012704 177777
7719 042176          LET R5 := #-1
      042176 012705 177777
7720 042202          INLINE <CMPNI>
      042202 076152
7721 042204          INLINE <.WORD TCMPI>
      042204 042320
7722 042206          INLINE <.WORD TCMPI*4>
      042206 042324
7723 042210          LET CC := PSW
      042210 016767 135562 136302
7724 042216          LET CC := CC CLR.BY #177760
      042216 042767 177760 136274
7725 042224          IF CC NE #4 THEN
      042224 026727 136270 000004
      042232 001402
7726 042234          CALL ERROR
      042234 004767 015440
7727 042240          ENDIF
      042240
      $51723:
7728 042240          IF R0 NE #-1 OR R1 NE #-1 OR R2 NE #-1 THEN
      042240 020027 177777
      042244 001006
      042246 020127 177777
      042252 001003

```

CMPN

```

042254 020227 177777                                CMP      R2,0-1
042260 001402                                BEQ      $51725
042262                                $51724:
7729 042262                                CALL ERROR
042262 004767 015412                                JSR      PC,ERROR
7130 042266                                ENDIF
042266                                $51725:
7731 042266                                IF R3 NE #-1 OR R4 NE #-1 OR R5 NE #-1 THEN
042266 020327 177777                                CMP      R3,0-1
042272 001006                                BNE      $51726
042274 020427 177777                                CMP      R4,0-1
042300 001003                                BNE      $51726
042302 020527 177777                                CMP      R5,0-1
042306 001402                                BEQ      $51727
042310                                $51726:
7732 042310                                CALL ERROR
042310 004767 015364                                JSR      PC,ERROR
7733 042314                                ENDIF
042314                                $51727:
7734 042314                                INLINE <JMP XASHN>
042314 000167 000016                                JMP XASHN
7735
7736                                ;+
7737                                ; INLINE TEST CASE
7738                                ; -
7739
7740 042320 010005                                TCMPI:  .WORD  010005                                ;SRC1 DATA TYPE = UNSIGNED ZONED, LENGTH = 5
7741 042322 042330                                .WORD  CMPN11                                ; "
7742 042324 010005                                .WORD  010005                                ;SRC2 DATA TYPE = UNSIGNED ZONED, LENGTH = 5
7743 042326 042330                                .WORD  CMPN11                                ; "
7744
7745 042330 001 002 003 CMPN11: .BYTE 001,002,003,004,065 ;DECIMAL STRING = 12345
042333 004 065
7746                                .EVEN
7747

```

ASHN

```

7749 .SBTTL ASHN
7750 ;*****
7751 ;
7752 ; ASHN TEST
7753 ;
7754 ;*****
7755
7756 042336 ROUTINE XASHN
7757 042336 XASHN:
7758 042336 000004 INLINE <SCOPE> SCOPE
7759 042340 000240 INLINE <NOP> NOP
7760 042342 012767 000020 136034 LET $TIMES := #20 ;DO 20 ITERATIONS MOV #20,$TIMES
7761 042350 016767 022540 136112 LET COUNT := TASH ;GET TEST COUNT MOV TASH,COUNT
7762 042356 012767 065116 136160 LET TSTCAS := #TASH+2 ;POINT TO FIRST TEST CASE MOV #TASH+2,TSTCAS
7763 042364 005067 136060 LET SIGNO := #0 CLR SIGNO
7764 042370 ;REPEAT
7765 042370 005067 136126 INLINE <#5056:> #5056:
7766 042374 005067 136064 LET CC.EXP := #0 CLR CC.EXP
7767 LET DAT TYP := #0 CLR DAT TYP
7768 ;*
7769 ; GET TEST DATA FROM TABLE
7770 ;-
7771
7772 042400 LET RO := TSTCAS
7773 042400 016700 136140 MOV TSTCAS,RO
7774 042404 012067 136152 LET DATA00 := (RO). ;SRC.LENGTH MOV (RO).,DATA00
7775 042410 012067 136150 LET DATA01 := (RO). ;DST.LENGTH MOV (RO).,DATA01
7776 042414 011067 136146 LET DATA02 := (RO) ;SHIFT.DSCR MOV (RO),DATA02
7777 042420 004767 014250 CALL CLRDSB JSR PC,CLRDSB
7778 ;*
7779 ; PUT TEST STRING INTO DSBUF1 WITHOUT THE SIGN
7780 ;-
7781
7782 042424 LET RO := DATA00 ;SRC.LENGTH MOV DATA00,RO
7783 042430 012701 064440 LET R1 := #DSBUF1 ;SRC.ADDRESS MOV #DSBUF1,R1
7784 042434 WHILE RO NE #0 DO
7785 042436 005700 ;$51732: TST RO
042440 001404 LET (R1). :B= #4 BEQ $51733

```



ASHN

7786	042440	112721	000004		LET R0 := R0 - #1	MOV B	#4.(R1).
	042444					DEC	R0
7787	042444	005300			ENDDO	BR	\$51732
	042446	000772				\$51733:	
	042450						
7788							
7789					;* ; LOAD DS.EXP AND CC.EXP WITH EXPECTED RESULTS. ;-		
7790							
7791							
7792							
7793					;GET SHIFT COUNT MAGNITUDE AND PUT IT INTO TEMPO		
7794							
7795	042450				LET TEMPO := DATA02		
	042450	016767	136112	136020		MOV	DATA02,TEMPO
7796	042456				IF #BIT07 SET IN TEMPO THEN	;NEGATIVE SHIFT COUNT	
	042456	032767	000200	136012		BIT	#BIT07,TEMPO
	042464	001402				BEQ	\$51734
7797	042466				LET TEMPO := NEGATE TEMPO		
	042466	005467	136004			NEG	TEMPO
7798	042472				ENDIF		
	042472					\$51734:	
7799	042472				LET TEMPO := TEMPO CLR.BY #177600		
	042472	042767	177600	135776		BIC	#177600,TEMPO
7800							
7801					;TAKE CARE OF EXPECTED NBIT		
7802							
7803	042500				IF SIGN0 EQ #-1 AND DATA00 NE #0 THEN		
	042500	026727	135744	177777		CMP	SIGN0,#-1
	042506	001006				BNE	\$51735
	042510	005767	136046			TST	DATA00
	042514	001403				BEQ	\$51735
7804	042516				LET CC.EXP := CC.EXP SET.BY #NBIT		
	042516	052767	000010	135776		BIS	#NBIT,CC.EXP
7805	042524				ENDIF		
	042524					\$51735:	
7806							
7807					;TAKE CARE OF ZERO LENGTH SOURCE OR DESTINATION		
7808							
7809	042524				IF DATA01 NE #0 THEN		
	042524	005757	136034			TST	DATA01
	042530	001414				BEQ	\$51736
7810	042532				LET R0 := #DS.EXP		
	042532	012700	064606			MOV	#DS.EXP,R0
7811	042536				INCR R1 FROM #1 TO DATA01 BY #1		
	042536	012701	000001			MOV	#1,R1
	042542	000401				BR	\$51737
	042544					\$51740:	
	042544	005201				INC	R1
	042546					\$51737:	
	042546	020167	136012			CMP	R1,DATA01
	042552	003003				BGT	\$51741
7812	042554				LET (R0).:B= #60		
	042554	112720	000060			MOV B	#60.(R0).
7813	042560				ENDINC		
	042560	000771				BR	\$51740
	042562					\$51741:	



ASHN

042714	005703						TST	R3
042716	101415						BLOS	\$51753
7837	042720				LET (R1) :B= #64			
7838	042720	112741	000064				MOVB	#64, -(R1)
	042724				IF R2 EQ #1 AND DATA02 HIS #3000 THEN			
	042724	020227	000001				CMP	R2, #1
	042730	001005					BNE	\$51754
	042732	026727	135630	003000			CMP	DATA02, #3000
	042740	103401					BLO	\$51754
7839	042742				LET (R1) :B= (R1) + #1		;ROUNDING	
	042742	105211					INCB	(R1)
7840	042744				ENDIF			
	042744						\$51754:	
7841	042744				LET R2 := R2 + #1			
7842	042746	005202			LET R3 := R3 #1		INC	R2
	042746	005303					DEC	R3
7843	042750				ENDDO		BR	\$51752
	042750	000757					\$51753:	
7844	042752				IF R3 EQ #0 AND R2 LE R0 THEN			
	042752	005703					TST	R3
	042754	001005					BNE	\$51755
	042756	020200					CMP	R2, R0
	042760	003003					BGT	\$51755
7845	042762				LET CC.EXP := CC.EXP SET.BY #VBIT			
	042762	052767	000002	135532			BIS	#VBIT, CC.EXP
7846	042770				ENDIF			
	042770						\$51755:	
7847	042770				ENDIF			
	042770						\$51745:	
7848								
7849					;TAKE CARE OF POSITIVE SHIFT			
7850								
7851	042770				IF #BIT07 NOTSETIN DATA02 THEN			
	042770	032767	000200	135570			BIT	#BIT07, DATA02
	042776	001072					BNE	\$51756
7852	043000				LET TEMP1 := DATA00 + TEMPO		;SRC.LEN + SHIFT COUNT	
	043000	016767	135556	135472			MOV	DATA00, TEMP1
	043006	066767	135464	135464			ADD	TEMPO, TEMP1
7853	043014				IF TEMP1 HI DATA01 THEN			
	043014	026767	135460	135542			CMP	TEMP1, DATA01
	043022	101403					BLOS	\$51757
7854	043024				LET CC.EXP := CC.EXP SET.BY #VBIT			
	043024	052767	000002	135470			BIS	#VBIT, CC.EXP
7855	043032				ENDIF			
	043032						\$51757:	
7856	043032				IF TEMPO HIS DATA01 THEN		;SHIFT COUNT >= DST.LEN	
	043032	026767	135440	135524			CMP	TEMPO, DATA01
	043040	103420					BLO	\$51760
7857	043042				LET CC.EXP := CC.EXP SET.BY #ZBIT			
	043042	052767	000004	135452			BIS	#ZBIT, CC.EXP
7858	043050				LET R1 := #DS.EXP			
	043050	012701	064606				MOV	#DS.EXP, R1
7859	043054				INCR R0 FROM #1 TO DATA01 BY #1			
	043054	012700	000001				MOV	#1, R0
	043060	000401					BR	\$51761

ASHN

	043062					\$51762:	INC	RO
	043062	005200				\$51761:	CMP	RO, DATA01
	043064						BGT	\$51763
	043064	020067	135474				MOVB	#60, (R1)+
7860	043070	003003					BR	\$51762
	043072			LET (R1)+ :B= #60				
	043072	112721	000060					
7861	043076			ENDINC				
	043076	000771				\$51763:		
	043100					;SHIFT COUNT <	DST.LEN	
7862	043100			ELSE		BR	\$51764	
	043100	000431				\$51760:		
	043102					MOV	#DS.EXP, R1	
7863	043102			LET R1 := #DS.EXP + DATA01		ADD	DATA01, R1	
	043102	012701	064606					
	043106	066701	135452					
7864	043112			INCR RO FROM #1 TO TEMPO BY #1		MOV	#1, RO	
	043112	012700	000001			BR	\$51765	
	043116	000401				\$51766:	INC	RO
	043120					\$51765:	CMP	RO, TEMPO
	043120	005200					BGT	\$51767
	043122							
	043122	020067	135350					
	043126	003003						
7865	043130			LET -(R1) :B= #60				
	043130	112741	000060					
7866	043134			ENDINC				
	043134	000771						
	043136					\$51767:		
7867	043136			LET RO := DATA00		;SRC.LEN		
	043136	016700	135420			MOV	DATA00, RO	
7868	043142			WHILE R1 HI #DS.EXP AND RO HI #0 DO				
	043142					\$51770:		
	043142	020127	064606			CMP	R1, #DS.EXP	
	043146	101406				BLOS	\$51771	
	043150	005700				TST	RO	
	043152	101404				BLOS	\$51771	
7869	043154			LET -(R1) :B= #64				
	043154	112741	000064					
7870	043160			LET RO := RO - #1				
	043160	005300						
7871	043162			ENDDO				
	043162	000767						
	043164					\$51771:		
7872	043164			ENDIF				
	043164					\$51764:		
7873	043164			ENDIF				
	043164					\$51756:		
7874	043164			ENDIF				
	043164					\$51744:		
7875	043164			IF #ZBIT SETIN CC.EXP THEN				
	043164	032767	000004			BIT	#ZBIT, CC.EXP	
	043172	001403	135330			BEQ	\$51772	
7876	043174			LET CC.EXP := CC.EXP CLR.BY #NBIT				
	043174	042767	000010			BIC	#NBIT, CC.EXP	
7877	043202			ENDIF				
	043202					\$51772:		

```

ASHN
7878
7879           ;SAVE THE FIRST AND LAST EXPECTED DST DIGITS
7880
7881 043202           LET DSTMP0 :B= DS.EXP
7882 043202 116767 021400 135344           MOVB   DS.EXP,DSTMP0
043210           LET RO := #DS.EXP + DATA01 #1           MOV    #DS.EXP,RO
043214 012700 064606           ADD    DATA01,RO
043220 005300           DEC    RO
7883 043222           LET DSTMP1 :B= (RO)           MOVB   (RO),DSTMP1
043222 111067 135327
7884
7885           ;+
7886           ; PERFORM THE ASHN INSTRUCTION WITH ALL SIX NUMERIC
7887           ; DATA TYPES. "DATTYP" WILL CONTAIN THE DATA TYPE
7888           ; CURRENTLY BEING TESTED.
7889           ;-
7890
7891           ;INCR DATTYP FROM #0 TO #50000 BY #10000
7892 043226           INLINE <CLR DATTYP>           CLR DATTYP
043226 005067 135232
7893 043232           INLINE <BR $5602>           BR $5602
043232 000403
7894 043234           INLINE <:$5601:>           $5601:
043234
7895 043234           LET DATTYP := DATTYP + #10000           ADD    #10000,DATTYP
043234 062767 010000 135222
7896 043242           INLINE <:$5602:>           $5602:
043242
7897 043242           IF DATTYP GT #50000 THEN           CMP    DATTYP,#50000
043242 026727 135216 050000           BLE    $51773
043250 003402
7898 043252           INLINE <JMP $5603>           JMP $5603
043252 000167 001554
7899 043256           ENDIF
043256
7900
7901           ;RESTORE FIRST AND LAST SRC DIGITS
7902 043256           IF DATA00 HI #0 THEN ;SRC.LEN > 0           TST   DATA00
043256 005767 135300           BLOS  $51774
043262 101411
7903 043264           LET DSBUF1 :B= #4           MOVB  #4,DSBUF1
043264 112767 000004 021146
7904 043272           LET RO := #DSBUF1-1 + DATA00           MOV   #DSBUF1-1,RO
043272 012700 064437           ADD  DATA00,RO
043276 066700 135260
7905 043302           LET (RO) :B= #4           MOVB  #4,(RO)
043302 112710 000004
7906 043306           ENDIF
043306
7907           ;RESTORE FIRST AND LAST DST DIGITS
7908 043306           IF DATA01 HI #0 THEN ;DST.LEN > 0           TST   DATA01
043306 005767 135252           BLOS  $51775
043312 101411
7909 043314           LET DS.EXP :B= DSTMP0           MOVB  DSTMP0,DS.EXP
043314 116767 135234 021264
7910 043322           LET RO := #DS.EXP-1 + DATA01

```

ASHN

```

043322 012700 064605
043326 066700 135232
7911 043332          LET (RO) :B= DSTMP1
043332 116710 135217
7912 043336          ENDIF
043336
7913
7914 043336          ;FINISH SETTING UP EXPECTED RESULTS ACCORDING TO DATA TYPE
043336 005767 135122          IF DATTYP EQ #0 THEN
043342 001067
7915 043344          LET DSBUF1-1 :B= #0
043344 105067 021067          ;RESTORE SRC AFTER LEADING SEPARATE
7916 043350          IF DATA00 HI #0 THEN
043350 005767 135206          ;SRC.LEN > 0
043354 101414
7917 043356          LET RO := #DSBUF1 1 + DATA00
043356 012700 064437
043362 066700 135174
7918 043366          IF SIGNO EQ #0 THEN
043366 005767 135056          ;SIGN IS +
043372 001003
7919 043374          LET (RO) :B= #64
043374 112710 000064
7920 043400          ELSE
043400 000402          ;SIGN IS
043402
7921 043402          LET (RO) :B= #164
043402 112710 000164
7922 043406          ENDIF
043406
7923 043406          ENDIF
043406
7924 043406          LET DS.EXP-1 :B= #0
043406 105067 021173          ;RESTORE DST AFTER LEADING SEPARATE
7925 043412          IF DATA01 HI #0 THEN
043412 005767 135146          ;DST.LEN > 0
043416 101441
7926 043420          LET RO := #DS.EXP-1 + DATA01
043420 012700 064605
043424 066700 135134
7927 043430          IF SIGNO EQ #0 OR DATA00 EQ #0 OR #ZBIT SETIN CC.EXP THEN
043430 005767 135014          ;SIGN IS +
043434 001407          TST SIGNO
043436 005767 135120          BEQ $52003
043442 001404          TST DATA00
043444 032767 000004 135050          BEQ $52003
043452 001403          BIT #ZBIT,CC.EXP
043454
7928 043454          LET (RO) :B= (RO) SET.BY #60
043454 152710 000060          ;SIGN IS
7929 043460          ELSE
043460 000402          ;SIGN IS
043462
7930 043462          LET (RO) :B= (RO) SET.BY #160
043462 152710 000160
7931 043466          ENDIF
043466

```

```

MOV #DS.EXP-1,RO
ADD DATA01,RO
MOV DSTMP1,(RO)
$51775:
TST DATTYP
BNE $51776
CLRB DSBUF1 1
TST DATA00
BLOS $51777
MOV #DSBUF1-1,RO
ADD DATA00,RO
TST SIGNO
BNE $52000
MOV #64,(RO)
BR $52001
$52000:
MOV #164,(RO)
$52001:
$51777:
CLRB DS.EXP-1
TST DATA01
BLOS $52002
MOV #DS.EXP-1,RO
ADD DATA01,RO
TST SIGNO
BEQ $52003
TST DATA00
BEQ $52003
BIT #ZBIT,CC.EXP
BEQ $52004
$52003:
BISB #60,(RO)
BR $52005
$52004:
BISB #160,(RO)
$52005:

```

ASHN

```

7932 043466          IFB DATA02 LO DATA00 AND #BIT07 NOTSETIN DATA02 AND SIGN0 EQ #-1 THEN
      043466 126767 135074 135066          CMPB DATA02,DATA00
      043474 103012          BHIS $52006
      043476 032767 000200 135062          BIT #BIT07,DATA02
      043504 001006          BNE $52006
      043506 026727 134736 177777          CMP SIGN0,#-1
      043514 001002          BNE $52006
7933 043516          LET (R0) :B= (R0) SET.BY #160          BISB #160,(R0)
      043516 152710 000160
7934 043522          ENDIF
      043522          $52006:
7935 043522          ENDIF
      043522          $52002:
7936 043522          ENDIF
      043522          $51776:
7937 043522          IF DATTYP EQ #10000 THEN          ;*** UNSIGNED ZONED ***
      043522 026727 134736 010000          CMP DATTYP,#10000
      043530 001025          BNE $52007
7938 043532          LET CC.EXP := CC.EXP CLR.BY #NBIT
      043532 042767 000010 134762          BIC #NBIT,CC.EXP
7939 043540          IF DATA00 HI #0 THEN          ;SRC.LEN > 0
      043540 005767 135016          TST DATA00
      043544 101406          BLOS $52010
7940 043546          LET R0 := #DSBUF1-1 + DATA00          MOV #DSBUF1-1,R0
      043546 012700 064437          ADD DATA00,R0
      043552 066700 135004
7941 043556          LET (R0) :B= #64          MOVB #64,(R0)
      043556 112710 000064
7942 043562          ENDIF
7943 043562          IF DATA01 HI #0 THEN          ;$52010:
      043562 005767 134776          ;DST.LEN > 0
      043566 101406          TST DATA01
7944 043570          LET R0 := #DS.EXP-1 + DATA01          BLOS $52011
      043570 012700 064605          MOV #DS.EXP-1,R0
      043574 066700 134764          ADD DATA01,R0
7945 043600          LET (R0) :B= (R0) SET.BY #60          BISB #60,(R0)
      043600 152710 000060
7946 043604          ENDIF
7947 043604          ENDIF          $52011:
7948 043604          IF DATTYP EQ #20000 THEN          ;$52007:
      043604 026727 134654 020000          ;*** TRAILING OVERPUNCH ***
      043612 001102          CMP DATTYP,#20000
7949 043614          IF DATA00 HI #0 THEN          ;SRC.LEN > 0
      043614 005767 134742          TST DATA00
      043620 101427          BLOS $52013
7950 043622          IF SIGN0 EQ #-1 AND #ZBIT NOTSETIN CC.EXP THEN
      043622 026727 134622 177777          CMP SIGN0,#-1
      043630 001007          BNE $52014
      043632 032767 000004 134662          BIT #ZBIT,CC.EXP
      043640 001003          BNE $52014
7951 043642          LET CC.EXP := CC.EXP SET.BY #NBIT ;RESTORE NBIT AFTER UNSIGNED ZONED
      043642 052767 000010 134652          BIS #NBIT,CC.EXP
7952 043650          ENDIF          $52014:
      043650

```

ASHN

7953	043650			LET R0 := #DSBUF1 1 + DATA00				
	043650	012700	064437				MOV	#DSBUF1-1,R0
	043654	066700	134702				ADD	DATA00,P0
7954	043660			IF SIGNO EQ #0 THEN	;SIGN IS +			
	043660	005767	134564				TST	SIGNO
	043664	001003					BNE	\$52015
7955	043666			LET (R0) :B= #104				
	043666	112710	000104				MOVB	#104,(R0)
7956	043672			ELSE	;SIGN IS -			
	043672	000402					BR	\$52016
	043674							\$52015:
7957	043674			LET (R0) :B= #115				
	043674	112710	000115				MOVB	#115,(R0)
7958	043700			ENDIF				\$52016:
	043700							
7959	043700			ENDIF				\$52013:
	043700							
7960	043700			IF DATA01 HI #0 THEN	;DST.LEN > 0			
	043700	005767	134660				TST	DATA01
	043704	101445					BLOS	\$52017
7961	043706			LET R0 := #DS.EXP-1 + DATA01				
	043706	012700	064605				MOV	#DS.EXP 1,R0
	043712	066700	134646				ADD	DATA01,R0
7962	043716			IF SIGNO EQ #0 OR DATA00 EQ #0 OR #ZBIT SETIN CC.EXP			THEN	;SIGN IS +
	043716	005767	134526				TST	SIGNO
	043722	001407					BEQ	\$52020
	043724	005767	134632				TST	DATA00
	043730	001404					BEQ	\$52020
	043732	032767	000004 134562				BIT	#ZBIT,CC.EXP
	043740	001402					BEQ	\$52021
	043742							\$52020:
7963	043742			LET R1 :B= #0				
	043742	105001					CLRB	R1
7964	043744			ELSE	;SIGN IS -			
	043744	000402					BR	\$52022
	043746							\$52021:
7965	043746			LET R1 :B= #10.				
	043746	112701	000012				MOVB	#10.,R1
7966	043752			ENDIF				\$52022:
	043752							
7967	043752			IFB DATA02 LO DATA00 AND #BIT07 NOTSETIN DATA02 AND	SIGNO EQ #-1 THEN			
	043752	126767	134610 134602				CMPB	DATA02,DATA00
	043760	103012					BHIS	\$52023
	043762	032767	000200 134576				BIT	#BIT07,DATA02
	043770	001006					BNE	\$52023
	043772	026727	134452 177777				CMP	SIGNO,# 1
	044000	001002					BNE	\$52023
7968	044002			LET R1 :B= #10.				
	044002	112701	000012				MOVB	#10.,R1
7969	044006			ENDIF				\$52023:
	044006							
7970	044006			LET R1 :B= R1 SET.BY (R0)				
	044006	151001					BISB	(R0),R1
7971	044010			LET R1 := R1 CLR.BY #177760				
	044010	042701	177760				BIC	#177760,R1
7972	044014			LET (R0) :B= OVPNCH(R1)				
	044014	116110	056330				MOVB	OVPNCH(R1),(R0)



ASHN

```

7973 044020          ENDIF
7974 044020          ENDIF
7975 044020          IF DATTYP EQ #30000 THEN
044020 026727 134440 030000
044026 001063
7976 044030          IF DATA00 HI #0 THEN
044030 005767 134526
044034 101412
7977 044036          IF SIGNO EQ #0 THEN
044036 005767 134406
044042 001004
7978 044044          LET DSBUF1 :B= #104
044044 112767 000104 020366
7979
7980 044052          ELSE
044052 000403
044054
7981 044054          LET DSBUF1 :B= #115
044054 112767 000115 020356
7982 044062          ENDIF
044062
7983 044062          ENDIF
044062
7984 044062          IF DATA01 HI #0 THEN
044062 005767 134476
044066 101443
7985 044070          IF SIGNO EQ #0 OR DATA00 EQ #0 OR #ZBIT SETIN CC.EXP
044070 005767 134354
044074 001407
044076 005767 134460
044102 001404
044104 032767 000004 134410
044112 001402
044114
7986 044114          LET R1 :B= #0
044114 105001
7987 044116          ELSE
044116 000402
044120
7988 044120          LET R1 :B= #10.
044120 112701 000012
7989 044124          ENDIF
044124
7990 044124          IFB DATA02 LO DATA00 AND #BIT07 NOTSETIN DATA02 AND SIGNO EQ #-1 THEN
044124 126767 134436 134430
044132 103012
044134 032767 000200 134424
044142 001006
044144 026727 134300 177777
044152 001002
7991 044154          LET R1 :B= #10.
044154 112701 000012
7992 044160          ENDIF
044160
7993 044160          LET R1 :B= R1 SET.BY DS.EXP

```

\$52017:

\$52012:

\*\*\* LEADING OVERPUNCH \*\*\*

CMP DATTYP,#30000  
BNE \$52024

;SRC.LEN > 0

TST DATA00  
BLOS \$52025

;SIGN IS +

TST SIGNO  
BNE \$52026

MOVB #104,DSBUF1

;SIGN IS -

BR \$52027

\$52026:

MOVB #115,DSBUF1

\$52027:

\$52025:

;DST.LEN > 0

TST DATA01  
BLOS \$52030  
THEN ;SIGN IS +  
TST SIGNO  
DEQ \$52031  
TST DATA00  
BEQ \$52031  
BIT #ZBIT,CC.EXP  
BEQ \$52032

\$52031:

CLRB R1

;SIGN IS

BR \$52033

\$52032:

MOVB #10.,R1

\$52033:

CMPB DATA02,DATA00  
BHIS \$52034  
BIT #BIT07,DATA02  
BNE \$52034  
CMP SIGNO,# 1  
BNE \$52034

\$52034:

MOVB #10.,R1

ASHN

```

7994 044160 156701 020422          LET R1 := R1 CLR.BY #177760          BISB  DS.EXP,R1
044164                                LET DS.EXP :B= OVPNCH(R1)          BIC   #177760,R1
7995 044164 042701 177760          LET DS.EXP :B= OVPNCH(R1)          MOVB  OVPNCH(R1),DS.EXP
044170 116167 056330 020410          ENDIF
7996 044176                                $52030:
044176                                ENDIF
7997 044176                                $52024:
044176                                ;*** TRAILING SEPARATE ***
7998 044176                                IF DATTYP EQ #40000 THEN          CMP   DATTYP,#40000
044176                                ;*** TRAILING SEPARATE ***          BNE   $52035
044204 026727 134262 040000
7999 001071
8000 ;PUT SIGN INTO SOURCE
8001
8002 044206          LET TEMPO := #DSBUF1 + DATA00          MOV   #DSBUF1,TEMPO
044206 012767 064440 134262          ADD   DATA00,TEMPO
044214 066767 134342 134254
8003 044222          IF SIGNO EQ #0 THEN          ;SIGN IS +
044222 005767 134222          ;SIGN IS +          TST   SIGNO
044226 001004          ;SIGN IS +          BNE   $52036
8004 044230          LET @TEMPO :B= #53          MOVB  #53,@TEMPO
044230 112777 000053 134240          ;SIGN IS -
8005 044236          ELSE          ;SIGN IS -          BR    $52037
044236 000403          ;SIGN IS -          $52036:
044240          LET @TEMPO :B= #55          MOVB  #55,@TEMPO
8006 044240 112777 000055 134230          $52037:
8007 044246          ENDIF
044246
8008 ;PUT EXPECTED SIGN INTO DS.EXP
8009
8010
8011 044246          LET TEMPO := #DS.EXP + DATA01          MOV   #DS.EXP,TEMPO
044246 012767 064606 134222          ADD   DATA01,TEMPO
044254 066767 134304 134214
8012 044262          IF SIGNO EQ #0 OR #ZBIT SETIN CC.EXP THEN          TST   SIGNO
044262 005767 134162          ;SIGN IS -          BEQ   $52040
044266 001404          ;SIGN IS -          BIT   #ZBIT,CC.EXP
044270 032767 000004 134224          ;SIGN IS -          BEQ   $52041
044276 001404          ;SIGN IS -          $52040:
044300          LET @TEMPO :B= #53          MOVB  #53,@TEMPO
8013 044300 112777 000053 134170          ELSE
8014 044306          ELSE          ;SIGN IS -          BR    $52042
044306 000403          ;SIGN IS -          $52041:
044310          LET @TEMPO :B= #55          MOVB  #55,@TEMPO
8015 044310 112777 000055 134160          $52042:
8016 044316          ENDIF          CMPB  DATA02,DATA00
044316          IFB DATA02 LO DATA00 AND #BIT07 NOTSETIN DATA02 AND SIGNO EQ # 1 THEN          BHIS  $52043
8017 044316 126767 134244 134236          BIT   #BIT07,DATA02
044324 103013          BNE   $52043
044326 032767 000200 134232          CMP   SIGNO,# 1
044334 001007
044336 026727 134106 177777

```

ASHN

8018	044344	001003			LET @TEMPO :B= #55	BNE	\$52043
	044346	112777	000055	134122		MOVB	#55,@TEMPO
8019	044354				ENDIF		
	044354						\$52043:
8020	044354				IF DATA01 EQ #0 THEN		;DST.LEN = 0
	044354	005767	134204			TST	DATA01
	044360	001003				BNE	\$52044
8021	044362				LET CC.EXP := CC.EXP CLR.BY #VBIT		
	044362	042767	000002	134132		BIC	#VBIT,CC.EXP
8022	044370				ENDIF		
	044370						\$52044:
8023	044370				ENDIF		
	044370						\$52035:
8024	044370				IF DATTYP EQ #50000 THEN		*** LEADING SEPARATE ***
	044370	026727	134070	050000		CMP	DATTYP,#50000
	044376	001075				BNE	\$52045
8025	044400				LET TEMPO := #DS.EXP + DATA01		
	044400	012767	064606	134070		MOV	#DS.EXP,TEMPO
	044406	066767	134152	134062		ADD	DATA01,TEMPO
8026	044414				LET @TEMPO :B= #0		;RESTORE DS.EXP FROM TRAILING SEPARATE
	044414	105077	134056			CLRB	@TEMPO
8027	044420				LET TEMPO := #DS.DST + DATA01		
	044420	012767	064544	134050		MOV	#DS.DST,TEMPO
	044426	066767	134132	134042		ADD	DATA01,TEMPO
8028	044434				LET @TEMPO :B= #0		;RESTORE DS.DST FROM TRAILING SEPARATE
	044434	105077	134036			CLRB	@TEMPO
8029							
8030					;PUT SIGN INTO SOURCE		
8031							
8032	044440				IF SIGN0 EQ #0 THEN		;SIGN IS +
	044440	005767	134074			TST	SIGN0
	044444	001004				BNE	\$52046
8033	044446				LET DSBUF1-1 :B= #53		
	044446	112767	000053	017763		MOVB	#53,DSBUF1 1
8034	044454				ELSE		;SIGN IS -
	044454	000403				BR	\$52047
	044456						\$52046:
8035	044456				LET DSBUF1-1 :B= #55		
	044456	112767	000055	017753		MOVB	#55,DSBUF1-1
8036	044464				ENDIF		
	044464						\$52047:
8037	044464				IF SIGN0 EQ #0 OR #ZBIT SETIN CC.EXP THEN		
	044464	005767	133760			TST	SIGN0
	044470	001404				BEQ	\$52050
	044472	032767	000004	134022		BIT	#ZBIT,CC.EXP
	044500	001404				BEQ	\$52051
	044502						\$52050:
8038	044502				LET DS.EXP-1 :B= #53		
	044502	112767	000053	020075		MOVB	#53,DS.EXP 1
8039	044510				ELSE		
	044510	000403				BR	\$52052
	044512						\$52051:
8040	044512				LET DS.EXP-1 :B= #55		
	044512	112767	000055	020065		MOVB	#55,DS.EXP 1
8041	044520				ENDIF		
	044520						\$52052:

ASHN

```

8042 044520          IFB DATA02 LO DATA00 AND #BIT07 NOTSETIN DATA02 AND SIGNO EQ #-1 THEN
      044520 126767 134042 134034          CMPB  DATA02,DATA00
      044526 103013          BHIS  $52053
      044530 032767 000200 134030          BIT   #BIT07,DATA02
      044536 001007          BNE   $52053
      044540 026727 133704 177777          CMP   SIGNO,# 1
      044546 001003          BNE   $52053
8043 044550          LET DS.EXP-1 :B= #55
      044550 112767 000055 020027          MOVB  #55,DS.EXP 1
8044 044556          ENDIF
      044556          ;$52053:
8045 044556          IF DATA01 EQ #0 THEN          ;DST.LEN = 0
      044556 005767 134002          TST   DATA01
      044562 001003          BNE   $52054
8046 044564          LET CC.EXP := CC.EXP CLR.BY #VBIT
      044564 042767 000002 133730          BIC   #VBIT,CC.EXP
8047 044572          ENDIF
      044572          ;$52054:
8048 044572          ENDIF          ;$52045:
      044572
8049
8050
8051          ;*
8052          ; SET UP THE GPR'S AND DO THE ASHN INSTRUCTION
8053          ; -
8054 044572          LET R0 := DATA00 SET.BY DATTYP
      044572 016700 133764          MOV   DATA00,R0
      044576 056700 133662          BIS   DATTYP,R0
8055 044602          LET R1 := #DSEUF1
      044602 012701 064440          MOV   #DSEUF1,R1
8056 044606          LET R2 := DATA01 SET.BY DATTYP
      044606 016702 133752          MOV   DATA01,R2
      044612 056702 133646          BIS   DATTYP,R2
8057 044616          LET R3 := #DS.DST
      044616 012703 064544          MOV   #DS.DST,R3
8058 044622          LET R4 := DATA02
      044622 016704 133740          MOV   DATA02,R4
8059 044626          LET R5 := #-1
      044626 012705 177777          MOV   #-1,R5
8060 044632          INLINE <ASHN>
      044632 076056          ASHN
8061
8062          ;*
8063          ; CHECK THE RESULTS AND REPORT ERRORS
8064          ; -
8065
8066 044634          LET CC := PSW
      044634 016767 133136 133656          MOV   PSW,CC
8067 044642          LET CC := CC CLR.BY #177760
      044642 042767 177760 133650          BIC   #177760,CC
8068 044650          IF CC NE CC.EXP THEN
      044650 026767 133644 133644          CMP   CC,CC.EXP
      044656 001402          BEQ   $52055
8069 044660          CALL ERROR
      044660 004767 013014          JSR   PC,ERROR
8070 044664          ENDIF
      044664          ;$52055:

```

ASHN

8071	044664				IF R0 NE #0 OR R1 NE #0 OR R4 NE #0 THEN			
	044664	005700					TST	R0
	044666	001004					BNE	\$52056
	044670	005701					TST	R1
	044672	001002					BNE	\$52056
	044674	005704					TST	R4
	044676	001402					BEQ	\$52057
	044700					\$52056:		
8072	044700				CALL ERROR			
	044700	004767	012774				JSR	PC.ERROR
8073	044704				ENDIF			
	044704					\$52057:		
8074	044704				LET TEMPO := DATA01 SET.BY DATTYP			
	044704	016767	133654	133564			MOV	DATA01,TEMPO
	044712	056767	133546	133556			BIS	DATTYP,TEMPO
8075	044720				IF R2 NE TEMPO THEN			
	044720	020267	133552				CMP	R2,TEMPO
	044724	001402					BEQ	\$52060
8076	044726				CALL ERROR			
	044726	004767	012746				JSR	PC.ERROR
8077	044732				ENDIF			
	044732					\$52060:		
8078	044732				IF R3 NE #DS.DST THEN			
	044732	020327	064544				CMP	R3,#DS.DST
	044736	001402					BEQ	\$52061
8079	044740				CALL ERROR			
	044740	004767	012734			\$52061:		
8080	044744				ENDIF			
	044744						JSR	PC.ERROR
8081					;COMPARE DS.EXP WITH DS.DST			
8082	044744				LET ERRFLG := #0			
	044744	005067	133506				CLR	ERRFLG
8083	044750				LET R0 := #41			
	044750	012700	000041				MOV	#41,R0
8084	044754				LET R1 := #DS.DST-1			
	044754	012701	064543				MOV	#DS.DST-1,R1
8085	044760				LET R2 := #DS.EXP-1			
	044760	012702	064605				MOV	#DS.EXP-1,R2
8086	044764				WHILE R0 NE #0 AND ERRFLG EQ #0 DO			
	044764					\$52062:		
	044764	005700					TST	R0
	044766	001412					BEQ	\$52063
	044770	005767	133462				TST	ERRFLG
	044774	001007					BNE	\$52063
8087	044776				IFB (R1)+ NE (R2)+ THEN			
	044776	122122					CMPB	(R1)+,(R2)+
	045000	001403					BEQ	\$52064
8088	045002				LET ERRFLG := #-1			
	045002	012767	177777	133446			MOV	#-1,ERRFLG
8089	045010				ENDIF			
	045010					\$52064:		
8090	045010				LET R0 := R0 #1			
	045010	005300					DEC	R0
8091	045012				ENDDO			
	045012	000764					BR	\$52062
	045014					\$52063:		
8092	045014				IF ERRFLG NE #0 THEN			

ASHN							
	045014	005767	133436			TST	ERRFLG
	045020	001402				BEQ	\$52065
8093	045022			CALL ERROR			
	045022	004767	012652			JSR	PC,ERROR
8094	045026			ENDIF			
	045026					\$52065:	
8095				;ENDINC			
8096	045026			INLINE <JMP \$5601>		JMP	\$5601
	045026	000167	176202				
8097	045032			INLINE <\$5603:>			\$5603:
	045032						
8098							
8099	045032			INCR RO FROM #DS.EXP-1 TO #DS.EXP+37 BY #1			
	045032	012700	064605			MOV	#DS.EXP-1,RO
	045036	000401				BR	\$52066
	045040					\$52067:	
	045040	005200				INC	RO
	045042					\$52066:	
	045042	020027	064645			CMP	RO,#DS.EXP+37
	045046	003002				BGT	\$52070
8100	045050			LET (RO) :B= #0		;CLEAR DS,DST	
	045050	105010				CLRB	(RO)
8101	045052			ENDINC			
	045052	000772				BR	\$52067
	045054					\$52070:	
8102	045054			LET SIGNO := COMP SIGNO		;REPEAT THE TEST CASE WITH A NEGATIVE	
	045054	005167	133370			COM	SIGNO
8103				; SIGN BEFORE MOVING ON TO THE NEXT ONE			
8104	045060			IF SIGNO EQ #0 THEN			
	045060	005767	133364			TST	SIGNO
	045064	001005				BNE	\$52071
8105	045066			LET TSTCAS := TSTCAS + #6			
	045066	062767	000006	133450		ADD	#6,TSTCAS
8106	045074			LET COUNT := COUNT - #1			
	045074	005367	133370			DEC	COUNT
8107	045100			ENDIF			
	045100					\$52071:	
8108				;UNTIL COUNT EQ #0			
8109	045100			IF COUNT NE #0 THEN			
	045100	005767	133364			TST	COUNT
	045104	001402				BEQ	\$52072
8110	045106			INLINE <JMP \$5056>		JMP	\$5056
	045106	000167	175256				
8111	045112			ENDIF			
	045112					\$52072:	
8112							
8113				;*			
8114				; DO ONE INLINE CASE - ASHNI			
8115				;-			
8116							
8117	045112			CALL CLRDSB			
	045112	004767	011556			JSR	PC,CLRDSB
8118							
8119				;LOAD SRC STRING INTO DSBUF1			
8120							
8121	045116			LET RO := #DSBUF1-1			
	045116	012700	064437			MOV	#DSBUF1-1,RO

ASHN

8122	045122			LET (R0)+ :B= #53	;LEADING SEPARATE SIGN OF .
	045122	112720	000053		MOVW #53,(R0).
8123	045126			LET (R0)+ :B= #9.	;PUT SRC MAGNITUDE OF 9999
	045126	112720	000011		MOVW #9.,(R0).
8124	045132			LET (R0)+ :B= #9.	; INTO DSBUF1
	045132	112720	000011		MOVW #9.,(R0).
8125	045136			LET (R0)+ :B= #9.	
	045136	112720	000011		MOVW #9.,(R0).
8126	045142			LET (R0) :B= #9.	
	045142	112710	000011		MOVW #9.,(R0)
8127	045146			LET R0 := #0	CLR R0
	045146	005000			
8128	045150			LET R1 := #0	CLR R1
	045150	005001			
8129	045152			LET R2 := #0	CLR R2
	045152	005002			
8130	045154			LET R3 := #0	CLR R3
	045154	005003			
8131	045156			LET R4 := #0	CLR R4
	045156	005004			
8132	045160			LET R5 := #0	CLR R5
	045160	005005			
8133	045162			INLINE <ASHNI>	ASHNI
	045162	076156			
8134	045164			INLINE <.WORD TASHNI>	.WORD TASHNI
	045164	045356			
8135	045166			INLINE <.WORD TASHNI+4>	.WORD TASHNI+4
	045166	045362			
8136	045170			INLINE <.WORD 000002>	;SHIFT COUNT - 2, RND DIGIT = 0
	045170	000002			.WORD 000002
8137	045172			LET CC := PSW	MOV PSW,CC
	045172	016767	132600	133320	
8138	045200			LET CC := CC CLR.BY #177760	BIC #177760,CC
	045200	042767	177760	133312	
8139	045206			IF CC NE #6 THEN	;VBIT SHOULD BE
	045206	026727	133306	000006	SET CC,#6
	045214	001402			BEQ \$52073
8140	045216			CALL ERROR	JSR PC,ERROR
	045216	004767	012456		
8141	045222			ENDIF	
	045222				\$52073:
8142	045222			IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN	TST R0
	045222	005700			BNE \$52074
	045224	001004			TST R1
	045226	005701			BNE \$52074
	045230	001002			TST R2
	045232	005702			BEQ \$52075
	045234	001402			
	045236				\$52074:
8143	045236			CALL ERROR	JSR PC,ERROR
	045236	004767	012436		
8144	045242			ENDIF	
	045242				\$52075:
8145	045242			IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN	TST R3
	045242	005703			BNE \$52076
	045244	001004			TST R4
	045246	005704			

```

ASHN
      045250 001002
      045252 005705
      045254 001402
      045256
8146 045256          CALL ERROR
      045256 004767 012416
8147 045262          ENDIF
      045262
8148          ;CHECK DS.DST
8149 045262          LET RO := #DS.DST
      045262 012700 064544
8150 045266          IFB (RO)+ NE #173 THEN
      045266 122027 000173
      045272 001402
8151 045274          CALL ERROR
      045274 004767 012400
8152 045300          ENDIF
      045300
8153 045300          IFB (RO)+ NE #60 THEN
      045300 122027 000060
      045304 001402
8154 045306          CALL ERROR
      045306 004767 012366
8155 045312          ENDIF
      045312
8156 045312          LET ERRFLG := #0
      045312 005067 133140
8157 045316          WHILE ERRFLG EQ #0 AND RO NE #DS.DST+37 DO
      045316
      045316 005767 133134
      045322 001013
      045324 020027 064603
      045330 001410
8158 045332          IFB (RO)+ NE #0 THEN
      045332 105720
      045334 001405
8159 045336          CALL ERROR
      045336 004767 012336
8160 045342          LET ERRFLG := #-1
      045342 012767 177777 133106
8161 045350          ENDIF
      045350
8162 045350          ENDDO
      045350 000762
      045352
8163 045352          INLINE <JMP CIS6END>
      045352 000167 000010
8164
8165          ;+
8166          ; INLINE TEST CASE
8167          ;-
8168
8169 045356 050004          TASHNI: .WORD 050004          ;SRC.DSCR, LEADING SEPARATE, SRC.LEN 4
8170 045360 064440          .WORD DSBUF1          ;SRC.DSCR
8171 045362 030002          .WORD 030002          ;DST.DSCR, LEADING OVERPUNCH, DST.LEN = 2
8172 045364 064544          .WORD DS.DST          ;DST.DSCR
8173
      BNE $52076
      TST R5
      BEQ $52077
$52076:
      JSR PC,ERROR
$52077:
      MOV #DS.DST,RO
      CMPB (RO)+,#173
      BEQ $52100
      JSR PC,ERROR
$52100:
      CMPB (RO)+,#60
      BEQ $52101
      JSR PC,ERROR
$52101:
      CLR ERRFLG
$52102:
      TST ERRFLG
      BNE $52103
      CMP RO,#DS.DST+37
      BEQ $52103
      TSTB (RO)+
      BEQ $52104
      JSR PC,ERROR
      MOV #1,ERRFLG
$52104:
      BR $52102
$52103:
      JMP CIS6END

```



ASHN

8174  
8175  
8176 045366  
045366  
8177  
8178

INLINE <CIS6END:>

CIS6END:

ASHN

8180  
8181  
8182  
8183  
8184  
8185  
8186  
8187  
8188  
8189  
8190  
8191  
8192  
8193  
8194  
8195  
8196  
8197  
8198  
8199  
8200  
8201  
8202  
8203  
8204  
8205  
8206  
8207  
8208  
8209  
8210  
8211  
8212  
8213  
8214  
8215

045366  
045366 000004  
045370  
045370 104406  
045372  
045372 017767 133030 133076  
045400 042767 177700 133070  
045406  
045406 005767 133064  
045412 001406  
045414  
045414 032767 000010 133054  
045422 001002  
045424  
045424 000167 005336  
045430  
045430  
045430  
045430 012767 000007 133110  
045436  
045436 005067 133014  
045442  
045442 012767 000340 132326  
045450  
045450 012767 000001 133074

```
*****
*****
;
;      C I S 7   M O D U L E
;
;  THIS MODULE TESTS THE PACKED DATA TYPE ARITHMETIC
;  INSTRUCTIONS: ADDP, SUBP, CMPP, ASHP.
;
;  --
;
;*****
;*****

.SBTTL  CIS7 INITIALIZATION
;*****
;
;  CIS7 INITIALIZATION
;
;*****

;IF SWR<5:0> ARE CLEARED OR IF SWR<3> IS SET THEN EXECUTE
; THE CIS7 MODULE; OTHERWISE, SKIP TO THE NEXT CIS MODULE.

      INLINE  <SCOPE>
                                SCOPE
      INLINE <CKSWR>
                                CKSWR
      LET TEMPO := @SWR CLR.BY #177700
                                MOV    @SWR,TEMPO
                                BIC    #177700,TEMPO
      IF TEMPO NE #0 THEN
                                TST    TEMPO
                                BEQ    $52105
                                IF #BIT3 NOTSETIN TEMPO THEN
                                        BIT    #BIT3,TEMPO
                                        BNE    $52106
                                INLINE  <JMP  CIS7END>
                                        JMP    CIS7END
      ENDIF
                                $52106:
      ENDIF
                                $52105:
      LET CTL := #7
                                ;CONTROL CHIP NUMBER
                                MOV    #7,CTL
      LET ERRFLG := #0
                                ;INITIALIZE ERROR FLAG
                                CLR    ERRFLG
      LET PSW := #340
                                MOV    #340,PSW
      LET ERR := #1
                                MOV    #1,ERR
```

ADDP

```

8217 .SBTTL ADDP
8218 ;*****
8219 ;
8220 ; ADDP TEST
8221 ;
8222 ;*****
8223
8224 ROUTINE XADDP
8225 045456 XADDP:
8226 045456 000004 INLINE <SCOPE> SCOPE
8227 045460 000240 INLINE <NOP> NOP
8228 045462 012767 000020 132714 LET $TIMES := #20 ;ITERATION COUNT
8229 045470 016767 017554 132772 LET COUNT := TADD ;# OF TEST CASES
8230 045476 012767 065252 133040 LET TSTCAS := #TADD+2 ;POINT TO FIRST TEST CASE
8231 ;
8232 ; CLR R0.EXP TO R3.EXP
8233 ;
8234
8235 045504 INCR R0 FROM #0 TO #6 BY #2
8236 045504 005000 CLR R0
8237 045506 000402 BR $52111
8238 045510 $52112: ADD #2,R0
8239 045514 062700 000002 $52111: CMP R0,#6
8240 045514 020027 000006 BGT $52113
8241 045520 003003
8242 045522 005060 000524 INLINE <CLR R0.EXP(R0)> CLR R0.EXP(R0)
8243 045526 000770 ENDINC BR $52112
8244 045530 $52113:
8245 ;REPEAT OUTER LOOP FOR TEST CASES
8246
8247 045530 INCR R0 FROM #0 TO #6 BY #2
8248 045530 004767 011140 CALL CLRDSB ;CLR DS BUFFERS
8249 045530 JSR PC,CLRDSB
8250 ;
8251 ; LOAD DATA0 AND CC.EXP FROM TEST CASE
8252 ;
8253 ;
8254 LET R0 := TSTCAS
8255 045534 016700 133004 MOV TSTCAS,R0
8256 045540 012067 133016 LET DATA0 := (R0)+ ;SRC1 LENGTH
8257 045544 012067 133014 LET DATA01 := (R0)+ ;POINTER TO SRC1 DATA
8258 045544 012067 133014 LET DATA02 := (R0)+ ;SRC2 LENGTH
8259 045550 012067 133012 MOV (R0)+,DATA00
8260 MOV (R0)+,DATA01
8261 MOV (R0)+,DATA02

```

```

ADDP
8252 045554          LET DATA03 := (R0)+          ;POINTER TO SRC2 DATA
      045554 012067 133010
8253 045560          LET DATA04 := (R0)+          ;DEST LENGTH
      045560 012067 133006
8254 045564          LET DATA05 := (R0)+          ;POINTER TO DS.EXP DATA (SUM)
      045564 012067 133004
8255 045570          LET CC.EXP := (R0)+          ;CC.EXP (SUM)
      045570 012067 132726
8256
8257
8258
8259
8260
8261 045574          LET R5 := #CHRSET          ;R5 IS OPERAND STACK FOR ROUTINE
      045574 012705 064036
8262 045600          CALL PACK IN <#DSBUF1,DATA01,DATA00> ;SRC1
      045600 010546
      045602 016745 132754
      045606 016745 132752
      045612 012745 064440
      045616 004767 011432
      045622 012605
8263 045624          CALL PACK IN <#DSBUF2,DATA03,DATA02> ;SRC2
      045624 010546
      045626 016745 132734
      045632 016745 132732
      045636 012745 064502
      045642 004767 011406
      045646 012605
8264 045650          CALL PACK IN <#DS.EXP,DATA05,DATA04> ;DEST
      045650 010546
      045652 016745 132714
      045656 016745 132712
      045662 012745 064606
      045666 004767 011362
      045672 012605
8265
8266
8267
8268
8269
8270 045674          LET R0 := DATA00 SET.BY #60000
      045674 016700 132662
      045700 052700 060000
8271 045704          LET R1 := #DSBUF1
      045704 012701 064440
8272 045710          LET R2 := DATA02 SET.BY #60000
      045710 016702 132652
      045714 052702 060000
8273 045720          LET R3 := #DSBUF2
      045720 012703 064502
8274 045724          LET R4 := DATA04 SET.BY #60000
      045724 016704 132642
      045730 052704 060000
8275 045734          LET R5 := #DS.DST
      045734 012705 064544
8276

```

ADDP

```

8277          ;+
8278          ;      FINISH SET UP & DO CIS INSTRUCTION
8279          ;-
8280
8281
8282 045740          LET R4.EXP := R4
8283 045740 010467 132570          MOV      R4,R4.EXP
8284 045744          LET R5.EXP := R5
8285 045744 010567 132566          MOV      R5,R5.EXP
8286 045750          INLINE <ADDP>
8287 045750 076070          ADDP
8288
8289          ;+
8290          ;      CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.
8291          ;-
8292 045752          LET CC := PSW
8293 045752 016767 132020 132540          MOV      PSW,CC
8294 045760          LET CC := CC CLR.BY #177760
8295 045760 042767 177760 132532          BIC      #177760,CC
8296 045766          IF CC NE CC.EXP THEN
8297 045766 026767 132526 132526          CMP      CC,CC.EXP
8298 045774 001402          BEQ      $52114
8299 045776          CALL ERROR
8300 045776 004767 011676          JSR      PC,ERROR
8301 046002          ENDIF
8302 046002          IF R0 NE R0.EXP THEN
8303 046002 020067 132516          CMP      R0,R0.EXP
8304 046006 001402          BEQ      $52115
8305 046010          CALL ERROR
8306 046010 004767 011664          JSR      PC,ERROR
8307 046014          ENDIF
8308 046014          IF R1 NE R1.EXP THEN
8309 046014 020167 132506          CMP      R1,R1.EXP
8310 046020 001402          BEQ      $52116
8311 046022          CALL ERROR
8312 046022 004767 011652          JSR      PC,ERROR
8313 046026          ENDIF
8314 046026          IF R2 NE R2.EXP THEN
8315 046026 020267 132476          CMP      R2,R2.EXP
8316 046032 001402          BEQ      $52117
8317 046034          CALL ERROR
8318 046034 004767 011640          JSR      PC,ERROR
8319 046040          ENDIF
8320 046040          IF R3 NE R3.EXP THEN
8321 046040 020367 132466          CMP      R3,R3.EXP
8322 046044 001402          BEQ      $52120
8323 046046          CALL ERROR
8324 046046 004767 011626          JSR      PC,ERROR
8325 046052          ENDIF
8326 046052          IF R4 NE R4.EXP THEN
8327 046052 020467 132456          CMP      R4,R4.EXP

```

ADDP

8308	046056	001402		CALL ERROR	BEQ	\$52121
	046060				JSR	PC,ERROR
8309	046064	004767	011614	ENDIF		
	046064				\$52121:	
8310	046064			IF R5 NE R5.EXP THEN		
	046064	020567	132446		CMP	R5,R5.EXP
	046070	001402			BEQ	\$52122
8311	046072			CALL ERROR		
	046072	004767	011602		JSR	PC,ERROR
8312	046076			ENDIF		
	046076				\$52122:	
8313						
8314						
8315				:+		
8316				:	CHECK RESULTS VS EXPECTED RESULTS	
8317				:-		
8318	046076			LET R0 := #DS.DST		
	046076	012700	064544		MOV	#DS.DST,R0
8319	046102			LET R1 := #DS.EXP		
	046102	012701	064606		MOV	#DS.EXP,R1
8320	046106			LET ERRFLG := #0		
	046106	005067	132344		CLR	ERRFLG
8321	046112			WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO		
	046112				\$52123:	
	046112	020027	064604		CMP	R0,#DS.DST+40
	046116	001410			BEQ	\$52124
	046120	005767	132332		TST	ERRFLG
	046124	001005			BNE	\$52124
8322	046126			IFB (R0)+ NE (R1)+ THEN		
	046126	122021			CMPB	(R0)+,(R1)+
	046130	001402			BEQ	\$52125
8323	046132			CALL ERROR		
	046132	004767	011542		JSR	PC,ERROR
8324	046136			ENDIF		
	046136				\$52125:	
8325	046136			ENDDO		
	046136	000765			BR	\$52123
	046140				\$52124:	
8326	046140			LET TSTCAS := TSTCAS + #22		
	046140	062767	000022 132376		ADD	#22,TSTCAS
8327	046146			LET COUNT := COUNT - #1		
	046146	005367	132316		DEC	COUNT
8328				;UNTIL COUNT EQ #0		
8329	046152			IF COUNT NE #0 THEN		
	046152	005767	132312		TST	COUNT
	046156	001402			BEQ	\$52126
8330	046160			INLINE <JMP \$5070>		
	046160	000167	177344		JMP	\$5070
8331	046164			ENDIF		
	046164				\$52126:	
8332						
8333				:+		
8334				:	INLINE CASE ADDPI, USING UNSIGNED PACKED DATA	
8335				:		
8336				:-		
8337	046164			CALL CLRDSB		

ADDP

```

      046164 004767 010504                                JSR    PC,CLRDSB
8338
8339
8340          ;+
8341          ;      LOAD DATAXX AND CC.EXP FROM TEST CASE
8342          ;
8343 046170          LET R0 := #TADD+2
      046170 012700 065252                                MOV    #TADD+2,R0
8344 046174          LET DATA00 := (R0)+                ;SRC1 LENGTH
      046174 012067 132362                                MOV    (R0)+,DATA00
8345 046200          LET DATA01 := (R0)+                ;POINTER TO SRC1 DATA
      046200 012067 132360                                MOV    (R0)+,DATA01
8346 046204          LET DATA02 := (R0)+                ;SRC2 LENGTH
      046204 012067 132356                                MOV    (R0)+,DATA02
8347 046210          LET DATA03 := (R0)+                ;POINTER TO SRC2 DATA
      046210 012067 132354                                MOV    (R0)+,DATA03
8348 046214          LET DATA04 := (R0)+                ;DEST LENGTH
      046214 012067 132352                                MOV    (R0)+,DATA04
8349 046220          LET DATA05 := (R0)+                ;POINTER TO DS.EXP DATA (SUM)
      046220 012067 132350                                MOV    (R0)+,DATA05
8350
8351          ;+
8352          ;      PACK BUFFERS FOR SRC1,SRC2, & DEST
8353          ;
8354          ;
8355 046224          LET R5 := #CHRSET                    ;R5 IS OPERAND STACK FOR ROUTINE
      046224 012705 064036                                MOV    #CHRSET,R5
8356 046230          CALL PACK IN <#DSBUF1,DATA01,DATA00> ;SRC1
      046230 010546
      046232 016745 132324                                MOV    R5,-(SP)
      046236 016745 132322                                MOV    DATA00,(R5)
      046242 012745 064440                                MOV    DATA01,-(R5)
      046246 004767 011002                                MOV    #DSBUF1,-(R5)
      046252 012605
      046254          CALL PACK IN <#DSBUF2,DATA03,DATA02> ;SRC2
      046254 010546
      046256 016745 132304                                MOV    R5,-(SP)
      046262 016745 132302                                MOV    DATA02,(R5)
      046266 012745 064502                                MOV    DATA03,(R5)
      046272 004767 010756                                MOV    #DSBUF2,-(R5)
      046276 012605
      046300          CALL PACK IN <#DS.EXP,DATA05,DATA04> ;DEST
      046300 010546
      046302 016745 132264                                MOV    R5,-(SP)
      046306 016745 132262                                MOV    DATA04,(R5)
      046312 012745 064606                                MOV    DATA05,(R5)
      046316 004767 010732                                MOV    #DS.EXP,(R5)
      046322 012605
      046324          CALL CLRREG                          ;ENSURE GPR'S 0-5 ARE ZERO
      046324 004767 010110                                JSR    PC,CLRREG
8361
8362 046330          INLINE <ADDP1>
      046330 076170
      046332          INLINE <.WORD TADDP1>
      046332 066136
      046334          INLINE <.WORD TADDP1+4>

```

ADDP

```

8365 046334 066142                               .WORD TADDPI+4
      046336                                .WORD TADDPI+10
8366 046340                                LET CC := PSW
      046340 016767 131432 132152             MOV     PSW,CC
8367 046346                                LET CC := CC CLR.BY #177760
      046346 042767 177760 132144           BIC     #177760,CC
8368 046354                                IF CC NE #0 THEN
      046354 005767 132140                   TST     CC
      046360 001402                           BEQ     $52127
8369 046362                                CALL ERROR
      046362 004767 011312                   JSR     PC.ERROR
8370 046366                                ENDIF
      046366                                $52127:
8371
8372
8373      ;+
8374      ;   ENSURE GPR'S STILL ZERO
8375      ;-
8376 046366                                IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN
      046366 005700                               TST     R0
      046370 001004                               BNE     $52130
      046372 005701                               TST     R1
      046374 001002                               BNE     $52130
      046376 005702                               TST     R2
      046400 001402                               BEQ     $52131
      046402                                $52130:
8377 046402                                CALL ERROR
      046402 004767 011272                   JSR     PC.ERROR
8378 046406                                ENDIF
      046406                                $52131:
8379 046406                                IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
      046406 005703                               TST     R3
      046410 001004                               BNE     $52132
      046412 005704                               TST     R4
      046414 001002                               BNE     $52132
      046416 005705                               TST     R5
      046420 001402                               BEQ     $52133
      046422                                $52132:
8380 046422                                CALL ERROR
      046422 004767 011252                   JSR     PC.ERROR
8381 046426                                ENDIF
      046426                                $52133:
8382
8383      ;+
8384      ;   CHECK RESULTS
8385      ;-
8386
8387 046426                                LET R0 := #DS.DST
      046426 012700 064544                               MOV     #DS.DST,R0
8388 046432                                LET R1 := #DS.EXP ;EXPECTED DATA
      046432 012701 064606                               MOV     #DS.EXP,R1
8389 046436                                LET R2 := TADDPI+10 CLR.BY #70000;GET STRING LENGTH
      046436 016702 017504                               MOV     TADDPI+10,R2
      046442 042702 070000                               BIC     #70000,R2
8390 046446                                INLINE <ASR R2> ;LENGTH/2 (PACKING FACTOR)
      046446 006202                               ASR     R2

```



ADDP

8391	046450		LET R2 := R2 * #DS.EXP		;R2 POINTS TO MSD/SIGN	
	046450	062702				ADD #DS.EYP,R2
8392	046454		LET (R2) :B= (R2) SET.BY #17 ;MAKE IT UNSIGNED			
	046454	152712				BISB #17,(R2)
8393	046460		LET ERRFLG := #0			
	046460	005067				CLR ERRFLG
8394	046464		WHILE RO NE #DS.DST+37 AND ERRFLG EQ #0 DO			
	046464				\$52134:	
	046464	020027				CMP RO,#DS.DST+37
	046470	001410				BEQ \$52135
	046472	005767				TST ERRFLG
	046476	001005				BNE \$52135
8395	046500		IFB (R0)+ NE (R1)+ THEN			
	046500	122021				CMPB (R0), (R1)
	046502	001402				BEQ \$52136
8396	046504		CALL ERROR			
	046504	004767				JSR PC,ERROR
8397	046510		ENDIF			
	046510				\$52136:	
8398	046510		ENDDO			
	046510	000765				BR \$52134
	046512				\$52135:	
8399						
8400						

SUBP

```

8402 .SBTTL SUBP
8403 ;*****
8404 ;
8405 ; SUBP TEST
8406 ;
8407 ;*****
8408
8409 ROUTINE XSUBP
8410 046512          XSUBP:
      046512          INLINE <SCOPE>          SCOPE
8411 046512 000004
      046514          INLINE <NOP>          NOP
8412 046514 000240
      046516          LET $TIMES := #20      ;ITERATION COUNT
8413 046516 012767 000020 131660          MOV      #20,$TIMES
      046524          LET COUNT := TADD      ;# OF TEST CASES
8414 046524 016767 016520 131736          MOV      TADD,COUNT
      046532          LET TSTCAS := #TADD*2  ;POINT TO FIRST TEST CASE
8415 046532 012767 065252 132004          MOV      #TADD*2,TSTCAS
8416
8417 ;*
8418 ; CLR R0.EXT TO R3.EXP
8419 ;
8420 INCR R0 FROM #0 TO #6 BY #2
8421 046540          CLR      R0
      046540 005000          BR      $52141
      046542 000402
      046544          —      $52142:
      046544 062700 000002          ADD      #2,R0
      046550          $52141:
      046550 020027 000006          CMP      R0,#6
      046554 003003          BGT      $52143
8421 046556          INLINE <CLR R0.EXP(R0)>
      046556 005060 000524          CLR R0.EXP(R0)
8422 046562          ENDINC
      046562 000770          BR      $52142
      046564          $52143:
8423
8424 ;REPEAT OUTER LOOP FOR TEST CASES
8425
8426 046564          INLINE <#5071:>
      046564          $5071:
8427 046564          CALL CLRDSB          ;CLR DS BUFFERS
      046564 004767 010104          JSR      PC,CLRDSB
8428
8429 ;*
8430 ; LOAD DATAXX AND CC.EXP FROM TEST CASE
8431 ;-
8432
8433 046570          LET R0 := TSTCAS
      046570 016700 131750          MOV      TSTCAS,R0
8434 046574          LET DATA00 := (R0)+   ;SRC1 LENGTH
      046574 012067 131762          MOV      (R0)+,DATA00
8435 046600          LET DATA01 := (R0)+   ;POINTER TO SRC1 DATA
      046600 012067 131760          MOV      (R0)+,DATA01
8436 046604          LET DATA02 := (R0)+   ;SRC2 LENGTH
      046604 012067 131756          MOV      (R0)+,DATA02

```

SUBP

```

8437 046610          LET DATA03 := (R0) ; POINTER TO SRC2 DATA          MOV (R0),DATA03
      046610 012067 131754
8438 046614          LET DATA04 := (R0) ; DEST LENGTH          MOV (R0),DATA04
      046614 012067 131752
8439 046620          LET DATA05 := (R0) ; POINTER TO DS.EXP DATA (SUM)  MOV (R0),DATA05
      046620 012067 131750
8440 046624          LET CC.EXP := (R0) ; CC.EXP (SUM)          MOV (R0),CC.EXP
      046624 012067 131672
8441 046630          LET DATA05 := (R0) ; POINTER TO DS.EXP DATA (DIFF)  MOV (R0),DATA05
      046630 012067 131740
8442 046634          LET CC.EXP := (R0) ; CC.EXP (DIFF)          MOV (R0),CC.EXP
      046634 012067 131662
8443
8444
8445      ;*          PACK BUFFERS FOR SRC1, SRC2, & DEST
8446      ;
8447      ;
8448 046640          LET R5 := #CHRSET ; R5 IS OPERAND STACK FOR ROUTINE  MOV #CHRSET,R5
      046640 012705 064036
8449 046644          CALL PACK IN <#DSBUF1,DATA01,DATA00> ; SRC1      MOV R5,(SP)
      046644 010546          MOV DATA00,-(R5)
      046646 016745 131710          MOV DATA01,-(R5)
      046652 016745 131706          MOV #DSBUF1,-(R5)
      046656 012745 064440          JSR PC,PACK
      046662 004767 010366          MOV (SP),R5
      046666 012605
8450 046670          CALL PACK IN <#DSBUF2,DATA03,DATA02> ; SRC2      MOV R5,-(SP)
      046670 010546          MOV DATA02,-(R5)
      046672 016745 131670          MOV DATA03,(R5)
      046676 016745 131666          MOV #DSBUF2,(R5)
      046702 012745 064502          JSR PC,PACK
      046706 004767 010342          MOV (SP),R5
      046712 012605
8451 046714          CALL PACK IN <#DS.EXP,DATA05,DATA04> ; DEST      MOV R5,(SP)
      046714 010546          MOV DATA04,(R5)
      046716 016745 131650          MOV DATA05,-(R5)
      046722 016745 131646          MOV #DS.EXP,-(R5)
      046726 012745 064606          JSR PC,PACK
      046732 004767 010316          MOV (SP),R5
      046736 012605
8452
8453
8454      ;*          LOAD GPR'S FROM DATAXX & ADD DATA TYPE
8455      ;
8456      ;
8457
8458 046740          LET R0 := DATA00 SET.BY #60000          MOV DATA00,R0
      046740 016700 131616          BIS #60000,R0
      046744 052700 060000
8459 046750          LET R1 := #DSBUF1          MOV #DSBUF1,R1
      046750 012701 064440
8460 046754          LET R2 := DATA02 SET.BY #60000          MOV DATA02,R2
      046754 016702 131606          BIS #60000,R2
      046760 052702 060000
8461 046764          LET R3 := #DSBUF2          MOV #DSBUF2,R3
      046764 012703 064502
8462 046770          LET R4 := DATA04 SET.BY #60000

```

SUBP

046770	016704	131576			MOV	DATA04,R4
046774	052704	060000			BIS	#60000,R4
8463	047000			LET R5 := #DS.DST		
	047000	012705	064544		MOV	#DS.DST,R5
8464						
8465				:+		
8466				;	FINISH SET UP & DO CIS INSTRUCTION	
8467				;-		
8468						
8469						
8470	047004			LET R4.EXP := R4		
	047004	010467	131524		MOV	R4,R4.EXP
8471	047010			LET R5.EXP := R5		
	047010	010567	131522		MOV	R5,R5.EXP
8472	047014			INLINE <SUBP>		
	047014	076071			SUBP	
8473						
8474				:+		
8475				;	CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.	
8476				;-		
8477						
8478	047016			LET CC := PSW		
	047016	016767	130754 131474		MOV	PSW,CC
8479	047024			LET CC := CC CLR.BY #177760		
	047024	042767	177760 131466		BIC	#177760,CC
8480	047032			IF CC NE CC.EXP THEN		
	047032	026767	131462 131462		CMP	CC,CC.EXP
	047040	001402			BEQ	\$52144
8481	047042			CALL ERROR		
	047042	004767	010632		JSR	PC,ERROR
8482	047046			ENDIF		
	047046				\$52144:	
8483	047046			IF R0 NE R0.EXP THEN		
	047046	020067	131452		CMP	R0,R0.EXP
	047052	001402			BEQ	\$52145
8484	047054			CALL ERROR		
	047054	004767	010620		JSR	PC,ERROR
8485	047060			ENDIF		
	047060				\$52145:	
8486	047060			IF R1 NE R1.EXP THEN		
	047060	020167	131442		CMP	R1,R1.EXP
	047064	001402			BEQ	\$52146
8487	047066			CALL ERROR		
	047066	004767	010606		JSR	PC,ERROR
8488	047072			ENDIF		
	047072				\$52146:	
8489	047072			IF R2 NE R2.EXP THEN		
	047072	020267	131432		CMP	R2,R2.EXP
	047076	001402			BEQ	\$52147
8490	047100			CALL ERROR		
	047100	004767	010574		JSR	PC,ERROR
8491	047104			ENDIF		
	047104				\$52147:	
8492	047104			IF R3 NE R3.EXP THEN		
	047104	020367	131422		CMP	R3,R3.EXP
	047110	001402			BEQ	\$52150
8493	047112			CALL ERROR		

SUBP

8494	047112	004767	010562			JSR	PC,ERROR
	047116			ENDIF			
8495	047116					\$52150:	
	047116			IF R4 NE R4.EXP THEN			
	047116	020467	131412			CMP	R4,R4.EXP
	047122	001402				BEQ	\$52151
8496	047124			CALL ERROR			
	047124	004767	010550			JSR	PC,ERROR
8497	047130			ENDIF			
	047130					\$52151:	
8498	047130			IF R5 NE R5.EXP THEN			
	047130	020567	131402			CMP	R5,R5.EXP
	047134	001402				BEQ	\$52152
8499	047136			CALL ERROR			
	047136	004767	010536			JSR	PC,ERROR
8500	047142			ENDIF			
	047142					\$52152:	
8501							
8502							
8503				;			
8504				;	CHECK RESULTS VS EXPECTED RESULTS		
8505				;			
8506				;-			
8507	047142			LET R0 := #DS.DST			
	047142	012700	064544			MOV	#DS.DST,R0
8508	047146			LET R1 := #DS.EXP			
	047146	012701	064606			MOV	#DS.EXP,R1
8509	047152			LET ERRFLG := #0			
	047152	005067	131300			CLR	ERRFLG
8510	047156			WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO			
	047156					\$52153:	
	047156	020027	064604			CMP	R0,#DS.DST+40
	047162	001410				BEQ	\$52154
	047164	005767	131266			TST	ERRFLG
	047170	001005				BNE	\$52154
8511	047172			IFB (R0)+ NE (R1)+ THEN			
	047172	122021				CMPB	(R0)+,(R1)+
	047174	001402				BEQ	\$52155
8512	047176			CALL ERROR			
	047176	004767	010476			JSR	PC,ERROR
8513	047202			ENDIF			
	047202					\$52155:	
8514	047202			ENDDO			
	047202	000765				BR	\$52153
	047204					\$52154:	
8515	047204			LET TSTCAS := TSTCAS + #22			
	047204	062767	000022 131332			ADD	#22,TSTCAS
8516	047212			LET COUNT := COUNT #1			
	047212	005367	131252			DEC	COUNT
8517				;UNTIL COUNT EQ #0			
8518	047216			IF COUNT NE #0 THEN			
	047216	005767	131246			TST	COUNT
	047222	001402				BEQ	\$52156
8519	047224			INLINE <JMP \$5071>			
	047224	000167	177334			JMP	\$5071
8520	047230			ENDIF			
	047230					\$52156:	

SUBP

```

8521
8522
8523      ;+      INLINE CASE   SUBPI, USING UNSIGNED PACKED DATA
8524      ;-
8525
8526 047230      CALL CLRDSB
      047230 004767 007440      JSR      PC,CLRDSB
8527
8528      ;+      LOAD DATAXX AND CC.EXP FROM TEST CASE
8529      ;-
8530
8531
8532 047234      LET R0 := #TADD+2
      047234 012700 065252      MOV      #TADD+2,R0
8533 047240      LET DATA00 := (R0)+      ;SRC1 LENGTH
      047240 012067 131316      MOV      (R0)+,DATA00
8534 047244      LET DATA01 := (R0)+      ;POINTER TO SRC1 DATA
      047244 012067 131314      MOV      (R0)+,DATA01
8535 047250      LET DATA02 := (R0)+      ;SRC2 LENGTH
      047250 012067 131312      MOV      (R0)+,DATA02
8536 047254      LET DATA03 := (R0)+      ;POINTER TO SRC2 DATA
      047254 012067 131310      MOV      (R0)+,DATA03
8537 047260      LET DATA04 := (R0)+      ;DEST LENGTH
      047260 012067 131306      MOV      (R0)+,DATA04
8538 047264      LET DATA05 := (R0)+      ;POINTER TO DS.EXP DATA (SUM)
      047264 012067 131304      MOV      (R0)+,DATA05
8539 047270      LET CC.EXP := (R0)+
      047270 012067 131226      MOV      (R0)+,CC.EXP
8540 047274      LET DATA05 := (R0)+      ;POINTER TO DS.EXP DATA (DIFFERENCE)
      047274 012067 131274      MOV      (R0)+,DATA05
8541
8542      ;+      PACK BUFFERS FOR SRC1, SRC2, & DEST
8543      ;-
8544
8545
8546 047300      LET R5 := #CHRSET      ;R5 IS OPERAND STACK FOR ROUTINE
      047300 012705 064036      MOV      #CHRSET,R5
8547 047304      CALL PACK IN <#DSBUF1,DATA01,DATA00> ;SRC1
      047304 010546      MOV      R5,(SP)
      047306 016745 131250      MOV      DATA00,(R5)
      047312 016745 131246      MOV      DATA01,(R5)
      047316 012745 064440      MOV      #DSBUF1,(R5)
      047322 004767 007726      JSR      PC,PACK
      047326 012605      MOV      (SP)+,R5
8548 047330      CALL PACK IN <#DSBUF2,DATA03,DATA02> ;SRC2
      047330 010546      MOV      R5,(SP)
      047332 016745 131230      MOV      DATA02,(R5)
      047336 016745 131226      MOV      DATA03,-(R5)
      047342 012745 064502      MOV      #DSBUF2,-(R5)
      047346 004767 007702      JSR      PC,PACK
      047352 012605      MOV      (SP)+,R5
8549 047354      CALL PACK IN <#DS.EXP,DATA05,DATA04> ;DEST
      047354 010546      MOV      R5,-(SP)
      047356 016745 131210      MOV      DATA04,(R5)
      047362 016745 131206      MOV      DATA05,(R5)
      047366 012745 064606      MOV      #DS.EXP,(R5)
      047372 004767 007656      JSR      PC,PACK

```

SUBP

8550	047376	012605			CALL CLRREG				MOV	(SP),R5
	047400								JSR	PC,CLRREG
8551	047400	004767	007034		INLINE <SUBPI>					
	047404	076171							SUBPI	
8552	047406	066136			INLINE <.WORD TADDPI>				.WORD	TADDPI
8553	047410	066142			INLINE <.WORD TADDPI+4>				.WORD	TADDPI+4
8554	047412	066146			INLINE <.WORD TADDPI+10>				.WORD	TADDPI+10
8555	047414				LET CC := PSW				MOV	PSW,CC
8556	047414	016767	130356	131076	LET CC := CC CLR.BY #177760				BIC	#177760,CC
8557	047422	042767	177760	131070	IF CC NE #0 THEN				TST	CC
	047430	005767	131064						BEQ	\$52157
	047434	001402			CALL ERROR				JSR	PC,ERROR
8558	047436	004767	010236		ENDIF					
8559	047442									\$52157:
8560										
8561					;++					
8562					;	ENSURE GPR'S STILL ZERO				
8563					;-					
8564										
8565	047442				IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN					
	047442	005700							TST	R0
	047444	001004							BNE	\$52160
	047446	005701							TST	R1
	047450	001002							BNE	\$52160
	047452	005702							TST	R2
	047454	001402							BEQ	\$52161
	047456									\$52160:
8566	047456				CALL ERROR				JSR	PC,ERROR
	047456	004767	010216		ENDIF					
8567	047462									\$52161:
	047462				IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN					
8568	047462	005703							TST	R3
	047464	001004							BNE	\$52162
	047466	005704							TST	R4
	047470	001002							BNE	\$52162
	047472	005705							TST	R5
	047474	001402							BEQ	\$52163
	047476									\$52162:
8569	047476				CALL ERROR				JSR	PC,ERROR
	047476	004767	010176		ENDIF					
8570	047502									\$52163:
	047502									
8571										
8572					;++					
8573					;	CHECK RESULTS				
8574					;-					
8575										
8576	047502				LET R0 := #DS.DST					

SUBP

8577	047502	012700	064544	LET R1 := #DS.EXP	;EXPECTED DATA	MOV	#DS.DST,R0
	047506					MOV	#DS.EXP,R1
8578	047512	012701	064606	LET R2 := TADDPI+10 CLR.BY #70000	;GET STRING LENGTH	MOV	TADDPI+10,R2
	047516	042702	070000			BIC	#70000,R2
8579	047522	016702	016430	INLINE <ASR R2>	;LENGTH/2 (PACKING FACTOR)	ASR	R2
	047522	006202				ADD	#DS.EXP,R2
8580	047524	062702	064606	LET R2 := R2 + #DS.EXP	;R2 POINTS TO MSD/SIGN	BISB	#17,(R2)
8581	047530	152712	000017	LET (R2) := (R2) SET.BY #17	;MAKE IT UNSIGNED	CLR	ERRFLG
8582	047534	005067	130716	LET ERRFLG := #0			
8583	047540			WHILE R0 NE #DS.DST+37 AND ERRFLG EQ #0 DO			
	047540					\$52164:	
	047540	020027	064603			CMP	R0,#DS.DST+37
	047544	001410				BEQ	\$52165
	047546	005767	130704			TST	ERRFLG
	047552	001005				BNE	\$52165
8584	047554			IFB (R0) NE (R1) THEN			
	047554	122021				CMPB	(R0), (R1)
	047556	001402				BEQ	\$52166
8585	047560			CALL ERROR		JSR	PC,ERROR
8586	047564	004767	010114	ENDIF			
	047564					\$52166:	
8587	047564			ENDDO			
	047564	000765				BR	\$52164
	047566					\$52165:	
8588							
8589							



CMPP

```

8591 .SBTTL CMPP
8592 ;*****
8593 ;
8594 ; CMPP TEST
8595 ;
8596 ;*****
8597
8598 047566 ROUTINE XCMPP
      047566
8599 047566          XCMPP:
      047566 000004          SCOPE
8600 047570          INLINE <NOP>
      047570 000240          NOP
8601 047572          LET $TIMES := #20          ;ITERATION COUNT
      047572 012767 000020 130604          MOV #20,$TIMES
8602 047600          LET COUNT := TCMP          ;GET TEST COUNT
      047600 016767 015042 130662          MOV TCMP,COUNT
8603 047606          LET TSTCAS := #TCMP+2          ;POINT TO FIRST TEST CASE
      047606 012767 064650 130730          MOV #TCMP+2,TSTCAS
8604 ;REPEAT
8605 047614          INLINE < $5072: >
      047614          $5072:
8606 047614          CALL CLRDSB
      047614 004767 007054          JSR PC,CLRDSB
8607 047620          LET CC.EXP := #0
      047620 005067 130676          CLR CC.EXP
8608 047624          LET SIGNO := #0
      047624 005067 130620          CLR SIGNO
8609 047630          LET DATTP := #0
      047630 005067 130630          CLR DATTP
8610
8611 ;+
8612 ; GET SOURCE DESCRIPTORS FROM TABLE
8613 ;-
8614
8615 047634          LET R0 := TSTCAS
      047634 016700 130704          MOV TSTCAS,R0
8616 047640          LET DATA0 := (R0)+          ;SRC1 LENGTH
      047640 012067 130716          MOV (R0)+,DATA0
8617 047644          LET DATA01 := (R0)+          ;SRC1 POINTER
      047644 012067 130714          MOV (R0)+,DATA01
8618 047650          LET DATA02 := (R0)+          ;SRC2 LENGTH
      047650 012067 130712          MOV (R0)+,DATA02
8619 047654          LET DATA03 := (R0)          ;SRC2 POINTER
      047654 011067 130710          MOV (R0),DATA03
8620
8621 ;+
8622 ; LOAD SRC1 AND SRC2 INTO DSBUF1 AND DSBUF2
8623 ; IN NON-PACKED FORMAT, FOR NOW.
8624 ;-
8625
8626 047660          LET R2 := DATA01 + #2          ;GET SRC1 ADDRESS
      047660 016702 130700          MOV DATA01,R2
      047664 062702 000002          ADD #2,R2
8627 047670          LET R3 := #DSBUF1
      047670 012703 064440          MOV #DSBUF1,R3
8628 047674          LET DSTMP0 :B= (R2)          ;SAVE SRC1 MSD

```

CMPP

```

8629 047674 111267 130654                                MCVB (R2),DSTMP0
      047700                                DECR R1 FROM DATA00 TO #0 BY #1
      047700 016701 130656                                MOV DATA00,R1
      047704 000401                                BR $52171
      047706                                $52172:
      047706 005301                                DEC R1
      047710                                $52171:
      047710 005701                                TST R1
      047712 002402                                BLT $52173
8630 047714                                LET (R3)+ :B= (R2)+
      047714 112223                                ENDDDEC
8631 047716                                ENDDDEC
      047716 000773
      047720
8632 047720                                LET DSTMP1 :B= -(R2)                                ;SAVE SRC1 LSD
      047720 114267 130631                                ;GET SRC2 ADDRESS
8633 047724                                LET R2 := DATA03 + #2
      047724 016702 130640                                MOV DATA03,R2
      047730 062702 000002                                ADD #2,R2
8634 047734                                LET R3 := #DSBUF2
      047734 012703 064502                                MOV #DSBUF2,R3
8635 047740                                LET DSTMP2 :B= (R2)                                ;SAVE SRC2 MSD
      047740 111267 130612                                MCVB (R2),DSTMP2
8636 047744                                DECR R1 FROM DATA02 TO #0 BY #1
      047744 016701 130616                                MOV DATA02,R1
      047750 000401                                BR $52174
      047752                                $52175:
      047752 005301                                DEC R1
      047754                                $52174:
      047754 005701                                TST R1
      047756 002402                                BLT $52176
8637 047760                                LET (R3)+ :B= (R2)+
      047760 112223                                ENDDDEC
8638 047762                                ENDDDEC
      047762 000773
      047764
8639 047764                                LET DSTMP3 :B= -(R2)                                ;SAVE SRC2 LSD
      047764 114267 130567                                MCVB (R2),DSTMP3
8640
8641
8642 ;*
8643 ; COMPARE SRC1 AND SRC2, AND SET N AND Z BITS
8644 ; IN CC.EXP ACCORDINGLY.
8645 ;-
8646 047770                                IF DATA00 EQ #0 AND DATA02 EQ #0 THEN ;SRC1 LENGTH AND SRC2 LENGTH = 0
      047770 005767 130566                                TST DATA00
      047774 001007                                BNE $52177
      047776 005767 130564                                TST DATA02
      050002 001004                                BNE $52177
8647 050004                                LET CC.EXP := CC.EXP SET.BY #ZBIT
      050004 052767 000004 130510                                BIS #ZBIT,CC.EXP
8648 050012                                ELSE
      050012 000424                                BR $52200
      050014                                $52177:
8649 050014                                IF DATA00 EQ #0 AND @DATA03 EQ #0 THEN ;SRC1 LENGTH = 0 AND SRC2 SIGN IS .
      050014 005767 130542                                TST DATA00
      050020 001007                                BNE $52201

```

CMPP

8650	050022	005777	130542			TST	@DATA03
	050026	001004				BNE	\$52201
	050030				LET CC.EXP := CC.EXP SET.BY #NBIT		
8651	050030	052767	000010	130464		BIS	#NBIT,CC.EXP
	050036				ELSE		
	050036	000412				BR	\$52202
	050040					\$52201:	
8652	050040				IF DATA02 EQ #0 AND @DATA01 EQ # 1 THEN	;SRC2 LENGTH = 0 & SRC1 SIGN IS	
	050040	005767	130522			TST	DATA02
	050044	001007				BNE	\$52203
	050046	027727	130512	177777		CMP	@DATA01,# 1
	050054	001003				BNE	\$52203
8653	050056				LET CC.EXP := CC.EXP SET.BY #NBIT		
	050056	052767	000010	130436		BIS	#NBIT,CC.EXP
8654	050064				ENDIF		
	050064					\$52203:	
8655	050064				ENDIF		
	050064					\$52202:	
8656	050064				ENDIF		
	050064					\$52200:	
8657	050064				IF DATA00 NE #0 AND DATA02 NE #0 THEN	;SRC1 LENGTH AND SRC2 LENGTH > 0	
	050064	005767	130472			TST	DATA00
	050070	001520				BEQ	\$52204
	050072	005767	130470			TST	DATA02
	050076	001515				BEQ	\$52204
8658	050100				IF @DATA01 NE @DATA03 THEN	;SRC1 SIGN AND SRC2 SIGN ARE DIFFERENT	
	050100	027777	130460	130462		CMP	@DATA01,@DATA03
	050106	001410				BEQ	\$52205
8659	050110				IF @DATA01 EQ #-1 THEN	.SRC1 SIGN IS	
	050110	027727	130450	177777		CMP	@DATA01,# 1
	050116	001003				BNE	\$52206
8660	050120				LET CC.EXP := CC.EXP SET.BY #NBIT		
	050120	052767	000010	130374		BIS	#NBIT,CC.EXP
8661	050126				ENDIF		
	050126					\$52206:	
8662	050126				ELSE	;SRC1 SIGN AND SRC2 SIGN ARE EQUAL	
	050126	000501				BR	\$52207
	050130					\$52205:	
8663	050130				LET SIGN0 := @DATA01	;SAVE THE COMMON SIGN	
	050130	017767	130430	130312		MOV	@DATA01,SIGN0
8664	050136				IF DATA00 LO DATA02 AND SIGN0 EQ #0 THEN	;SRC1 LENGTH < SRC2 LENGTH AND SIGNS	
ARE	050136	026767	130420	130422		CMP	DATA00,DATA02
	050144	103007				BHIS	\$52210
	050146	005767	130276			TST	SIGN0
	050152	001004				BNE	\$52210
8665	050154				LET CC.EXP := CC.EXP SET.BY #NBIT		
	050154	052767	000010	130340		BIS	#NBIT,CC.EXP
8666	050162				ELSE		
	050162	000413				BR	\$52211
	050164					\$52210:	
8667	050164				IF DATA00 HI DATA02 AND SIGN0 EQ #-1 THEN	;SRC1 LENGTH > SRC2 LENGTH AND S	
IGNS ARE	050164	026767	130372	130374		CMP	DATA00,DATA02
	050172	101407				BLOS	\$52212
	050174	026727	130250	177777		CMP	SIGN0,# 1
	050202	001003				BNE	\$52212
8668	050204				LET CC.EXP := CC.EXP SET.BY #NBIT		
	050204	052767	000010	130310		BIS	#NBIT,CC.EXP

CMPP

```

8669 050212                ENDIF
      050212                $52212:
8670 050212                ENDIF
      050212                $52211:
8671 050212                IF DATA00 EQ DATA02 THEN ;SRC1 LENGTH AND SRC2 LENGTH ARE EQUAL
      050212 026767 130344 130346                ;SRC1 LENGTH AND SRC2 LENGTH ARE EQUAL
      050220 001044                CMP DATA00,DATA02
      050220 001044                BNE $52213
8672
8673
8674 ; COMPARE SRC1 AND SRC2 BYTE BY BYTE
8675 ;
8676 ;
8677 050222                LET R0 := DATA00 ;SRC BUFFER LENGTH
      050222 016700 130334                ;SRC BUFFER LENGTH
      050222 016700 130334                MOV DATA00,R0
8678 050226                LET R1 := #DSBUF1 ;SRC1 BUFFER
      050226 012701 064440                ;SRC1 BUFFER
      050226 012701 064440                MOV #DSBUF1,R1
8679 050232                LET R2 := #DSBUF2 ;SRC2 BUFFER
      050232 012702 064502                ;SRC2 BUFFER
      050232 012702 064502                MOV #DSBUF2,R2
8680 050236                WHILEB R0 NE #0 AND (R1)+ EQ (R2)+ DO
      050236                $52214:
      050236 105700                TSTB R0
      050240 001404                BEQ $52215
      050242 022122                CMP (R1)+,(R2)+
      050244 001002                BNE $52215
8681 050246                LET R0 := R0 - #1 ;DECREMENT COUNTER
      050246 005300                DEC R0
8682 050250                ENDDO
      050250 000772                BR $52214
      050252                $52215:
8683 050252                IFB -(R1) NF -(R2) THEN ;SOURCES WERE NOT
      050252 124142                CMPB (R1),(R2)
      050254 001423                BEQ $52216
8684 050256                IFB (R1) LO (R2) AND SIGN0 EQ #0 THEN
      050256 121112                CMPB (R1),(R2)
      050260 103007                BHIS $52217
      050262 005767 130162                TST SIGN0
      050266 001004                BNE $52217
8685 050270                LET CC.EXP := CC.EXP SET.BY #NBIT
      050270 052767 000010 130224                BIS #NBIT,CC.EXP
8686 050276                ELSE
      050276 000411                BR $52220
      050300                $52217:
8687 050300                IFB (R1) HI (R2) AND SIGN0 EQ #-1 THEN
      050300 121112                CMPB (R1),(R2)
      050302 101407                BLOS $52221
      050304 026727 130140 177777                CMP SIGN0,#1
      050312 001003                BNE $52221
8688 050314                LET CC.EXP := CC.EXP SET.BY #NBIT
      050314 052767 000010 130200                BIS #NBIT,CC.EXP
8689 050322                ENDIF
      050322                $52221:
8690 050322                ENDIF
      050322                $52220:
8691 050322                ELSE ;SOURCES WERE EQUAL
      050322 000403                BR $52222
      050324                $52216:
8692 050324                LET CC.EXP := CC.EXP SET.BY #ZBIT

```

CMPP

8693	050324	052767	000004	130170			BIS	#ZBIT,CC.EXP
	050332				ENDIF			
	050332						\$52222:	
8694	050332				ENDIF			
	050332						\$52213:	
8695	050332				ENDIF			
	050332						\$52207:	
8696	050332				ENDIF			
	050332						\$52204:	
8697								
8698								
8699					;			
8700					;			
8701					;			
8702					;			
8703					;			
8704					;			
8705	050332				CALL CLRDSB			
	050332	004767	006336				JSR	PC,CLRDSB
8706					;			
8707	050336				LOAD DSBUF1 WITH SRC1 DECIMAL STRING AND SIGN, IN PACKED FORMAT.			
	050336	012700	064440		LET R0 := #DSBUF1		MOV	#DSBUF1,R0
8708	050342				LET R1 := DATA01 + #2		MOV	DATA01,R1
	050342	016701	130216				ADD	#2,R1
	050346	062701	000002					
8709	050352				LET R2 := DATA00	;		
	050352	016702	130204				MOV	DATA00,R2
8710	050356				IF R2 EQ #0 THEN			
	050356	005702					TST	R2
	050360	001011					BNE	\$52223
8711	050362				IF @DATA01 EQ #0 THEN	;		
	050362	005777	130176				TST	@DATA01
	050366	001003					BNE	\$52224
8712	050370				LET (R0) :B= #14		MOVB	#14,(R0)
	050370	112710	000014					
8713	050374				ELSE	;		
	050374	000402					BR	\$52225
	050376							
8714	050376				LET (R0) :B= #15			
	050376	112710	000015				MOVB	#15,(R0)
8715	050402				ENDIF			
	050402						\$52225:	
8716	050402				ELSE			
	050402	000434					BR	\$52226
	050404							
8717	050404				IF #BIT00 NOTSETIN R2 THEN	;		
	050404	032702	000001					
	050410	001002					BIT	#BIT00,R2
	050412						BNE	\$52227
8718	050412				LET (R0)+ :B= (R1)+	;		
	050412	112120					MOVB	(R1)+,(R0)+
8719	050414				LET R2 := R2 - #1	;		
	050414	005302					DEC	R2
8720	050416				ENDIF			
	050416						\$52227:	
8721	050416				WHILE R2 NE #1 DO			
	050416						\$52230:	
	050416	020227	000001				CMP	R2,#1

CMPP

```

      050422 001410                                BEQ      $52231
8722 050424                                LET R3 :B= (R1) ;GET A BCD
      050424 112103                                MOV      (R1),R3
8723 050426                                INLINE <ASH #4,R3> ;SHIFT IT TO THE HIGH NIBBLE
      050426 072327 000004                                ASH #4,R3
8724 050432                                LET R3 :B= R3 SET.BY (R1) ;PUT NEXT BCD INTO LOWER NIBBLE
      050432 152103                                BIS      (R1),R3
8725 050434                                LET (R0) :B= R3
      050434 110320                                MOV      R3,(R0)
8726 050436                                LET R2 := R2 - #2 ;DECREMENT COUNTER
      050436 162702 000002                                SUB      #2,R2
8727 050442                                ENDDO
      050442 000765                                $52231:
      050444                                LET R3 :B= (R1) ;GET LAST BCD
8728 050444                                MOV      (R1),R3
      050444 111103                                INLINE <ASH #4,R3>
8729 050446                                IF @DATA01 EQ #0 THEN ;SIGN IS +
      050446 072327 000004                                ASH #4,R3
      050452 005777 130106                                TST      @DATA01
      050456 001003                                BNE      $52232
8731 050460                                LET R3 :B= R3 SET.BY #14
      050460 152703 000014                                BIS      #14,R3
8732 050464                                ELSE ;SIGN IS -
      050464 000402                                BR       $52233
      050466                                LET R3 :B= R3 SET.BY #15
8733 050466                                $52232:
      050466 152703 000015                                BIS      #15,R3
8734 050472                                ENDIF
      050472                                LET (R0) :B= R3 ;PUT MSD AND SIGN INTO THE BUFFER
8735 050472                                $52233:
      050472 110310                                MOV      R3,(R0)
8736 050474                                ENDIF
      050474                                $52226:
8737 ;LOAD DSBUF2 WITH SRC2 DECIMAL STRING AND SIGN, IN PACKED FORMAT.
8738 050474                                LET R0 := #DSBUF2
      050474 012700 064502                                MOV      #DSBUF2,R0
8739 050500                                LET R1 := DATA03 + #2
      050500 016701 130064                                MOV      DATA03,R1
      050504 062701 000002                                ADD      #2,R1
8740 050510                                LET R2 := DATA02 ;DECIMAL STRING LENGTH
      050510 016702 130052                                MOV      DATA02,R2
8741 050514                                IF R2 EQ #0 THEN ;LENGTH = 0
      050514 005702                                TST      R2
      050516 001011                                BNE      $52234
8742 050520                                IF @DATA03 EQ #0 THEN ;SIGN IS +
      050520 005777 130044                                TST      @DATA03
      050524 001003                                BNE      $52235
8743 050526                                LET (R0) :B= #14
      050526 112710 000014                                MOV      #14,(R0)
8744 050532                                ELSE ;SIGN IS -
      050532 000402                                BR       $52236
      050534                                LET (R0) :B= #15
8745 050534                                $52235:
      050534 112710 000015                                MOV      #15,(R0)
8746 050540                                ENDIF
      050540                                $52236:

```

CMPP

```

8747 050540          ELSE
      050540 000434
      050542
8748 050542          IF #BIT00 NOTSET IN R2 THEN          ;LENGTH IS EVEN AND > 0
      050542 032702 000001                                ;$52234:
      050546 001002                                BIT      #BIT00,R2
8749 050550          LET (R0) := (R1) := ;MOVE THE EVEN NIBBLE                                ;$52240:
      050550 112120                                BNE      $52240
8750 050552          LET R2 := R2 - #1                    ;DECREMENT COUNTER                                ;$52241:
      050552 005302                                MOV      (R1), (R0)
8751 050554          ENDIF
      050554
8752 050554          WHILE R2 NE #1 DO
      050554 020227 000001                                ;$52241:
      050560 001410                                CMP      R2, #1
8753 050562          LET R3 := (R1) := ;GET A BCD                                BEQ      $52242
      050562 112103                                MOV      (R1), R3
8754 050564          " INE <ASH #4,R3> ;SHIFT IT TO THE HIGH NIBBLE                                ;$52242:
      050564 072327 000004                                MOV      (R1), R3
8755 050570          LEI R3 := R3 SET.BY (R1) := ;PUT NEXT BCD INTO LOWER NIBBLE                                ;$52243:
      050570 152103                                ASH     #4,R3
8756 050572          LET (R0) := R3
      050572 110320                                BIS     (R1), R3
8757 050574          LET R2 := R2 - #2                    ;DECREMENT COUNTER                                MOV      R3, (R0)
      050574 162702 000002                                SUB     #2,R2
8758 050600          ENDDO
      050600 000765                                BR      $52241
      050602
8759 050602          LET R3 := (R1)
      050602 111103                                ;$52242:
8760 050604          INLINE <ASH #4,R3>                                MOV      (R1), R3
      050604 072327 000004                                ASH     #4,R3
8761 050610          IF @DATA03 EQ #0 THEN ;SIGN IS +                                ;$52243:
      050610 005777 127754                                TST     @DATA03
      050614 001003                                BNE     $52243
8762 050616          LET R3 := R3 SET.BY #14
      050616 152703 000014                                BIS     #14,R3
8763 050622          ELSE ;SIGN IS -
      050622 000402                                BR      $52244
      050624
8764 050624          LET R3 := R3 SET.BY #15
      050624 152703 000015                                ;$52243:
      050630          ENDIF                                BIS     #15,R3
8765 050630
      050630
8766 050630          LET (R0) := R3 ;PUT MSD AND SIGN INTO THE BUFFER                                ;$52244:
      050630 110310                                MOV      R3, (R0)
8767 050632          ENDIF
      050632
8768
8769
8770          ;*
8771          ; LOAD THE GPR'S AND DO THE CMPP INSTRUCTION
8772          ;*
8773 050632          LET R0 := #60000 SET.BY DATA00 ;SRC1.DSCR
      050632 012700 060000                                MOV      #60000,R0
      050636 056700 127720                                BIS     DATA00,R0

```

## CMPP

```

8774 050642          LET R1 := #DSBUF1          ; "
      050642 012701 064440
8775 050646          LET R2 := #60000 SET.BY DATA02      ;SRC2.DSCR
      050646 012702 060000
      050652 056702 127710
8776 050656          LET R3 := #DSBUF2          ; "
      050656 012703 064502
8777 050662          LET R4 := #0
      050662 005004
8778 050664          LET R5 := #0
      050664 005005
8779 050666          INLINE <CMPP>
      050666 076072
8780
8781
8782          ;+
8783          ; CHECK THE RESULTS AND REPORT ANY ERRORS
8784          ;-
8785 050670          LET CC := PSW
      050670 016767 127102 127622
8786 050676          LET CC := CC CLR.BY #177760
      050676 042767 177760 127614
8787 050704          IF CC NE CC.EXP THEN
      050704 026767 127610 127610
      050712 001402
      CALL ERROR
8788 050714          CALL ERROR
      050714 004767 006760
8789 050720          ENDIF
      050720
8790 050720          IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN
      050720 005700
      050722 001004
      050724 005701
      050726 001002
      050730 005702
      050732 001402
      050734
      CALL ERROR
8791 050734          CALL ERROR
      050734 004767 006740
8792 050740          ENDIF
      050740
8793 050740          IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
      050740 005703
      050742 001004
      050744 005704
      050746 001002
      050750 005705
      050752 001402
      050754
      CALL ERROR
8794 050754          CALL ERROR
      050754 004767 006720
8795 050760          ENDIF
      050760
8796 050760          LET TSTCAS := TSTCAS + #10
      050760 062767 000010 127556
8797 050766          LET COUNT := COUNT #1
      050766 005367 127476

```

MOV #DSBUF1,R1

MOV #60000,R2  
BIS DATA02,R2

MOV #DSBUF2,R3

CLR R4

CLR R5

CMPP

MOV PSW,CC

BIC #177760,CC

CMP CC,CC.EXP  
BEQ \$52245

JSR PC.ERROR

\$52245:

TST R0  
BNE \$52246  
TST R1  
BNE \$52246  
TST R2  
BEQ \$52247

\$52246:

JSR PC.ERROR

\$52247:

TST R3  
BNE \$52250  
TST R4  
BNE \$52250  
TST R5  
BEQ \$52251

\$52250:

JSR PC.ERROR

\$52251:

ADD #10,TSTCAS

DEC COUNT



CMPP

```

8798 ;UNTIL COUNT EQ #0
8799 050772 IF COUNT NE #0 THEN
      050772 005767 127472
      050776 001402
8800 051000          INLINE <JMP $5072>
      051000 000167 176610
8801 051004          ENDIF
      051004
      $52252:
8802
8803
8804 ; DO ONE INLINE CASE  CMPP.  USE UNSIGNED PACKED DATA TYPE.
8805 ; -
8806
8807 051004          LET R0 := #-1
      051004 012700 177777
8808 051010          LET R1 := #-1
      051010 012701 177777
8809 051014          LET R2 := #-1
      051014 012702 177777
8810 051020          LET R3 := #-1
      051020 012703 177777
8811 051024          LET R4 := #-1
      051024 012704 177777
8812 051030          LET R5 := #-1
      051030 012705 177777
8813 051034          INLINE <CMPP>
      051034 076172
8814 051036          INLINE <.WORD TCMPP>
      051036 051152
8815 051040          INLINE <.WORD TCMPP+4>
      051040 051156
8816 051042          LET CC := PSW
      051042 016767 126730 127450
8817 051050          LET CC := CC CLR.BY #177760
      051050 042767 177760 127442
8818 051056          IF CC NE #4 THEN
      051056 026727 127436 000004
      051064 001402
8819 051066          CALL ERROR
      051066 004767 006606
8820 051072          ENDIF
      051072
      $52253:
8821 051072          IF R0 NE #-1 OR R1 NE # 1 OR R2 NE #-1 THEN
      051072 020027 177777
      051076 001006
      051100 020127 177777
      051104 001003
      051106 020227 177777
      051112 001402
      051114
      $52254:
8822 051114          CALL ERROR
      051114 004767 006560
8823 051120          ENDIF
      051120
      $52255:
8824 051120          IF R3 NE # 1 OR R4 NE #-1 OR R5 NE #-1 THEN
      051120 020327 177777
      051124 001006

```

```

TST      COUNT
BEQ      $52252
JMP      $5072
MOV      #-1,R0
MOV      # 1,R1
MOV      # 1,R2
MOV      # 1,R3
MOV      #-1,R4
MOV      # 1,R5
CMPP
.WORD   TCMPP
.WORD   TCMPP+4
MOV      PSW,CC
BIC      #177760,CC
CMP      CC,#4
BEQ      $52253
JSR      PC,ERROR
CMP      R0,#-1
BNE      $52254
CMP      R1,#-1
BNE      $52254
CMP      R2,# 1
BEQ      $52255
JSR      PC,ERROR
CMP      R3,# 1
BNE      $52256

```

CMPP

```

051126 020427 177777          CMP      R4,0-1
051132 001003                  BNE      $52256
051134 020527 177777          CMP      R5,0-1
051140 001402                  BEQ      $52257
051142                          $52256:
8825 051142          CALL ERROR
051142 004767 006532          JSR      PC,ERROR
8826 051146          ENDIF
051146                  $52257:
8827 051146          INLINE <JMP XASHP>
051146 000167 000014          JMP XASHP
8828
8829                      ;+
8830                      ;INLINE TEST CASE
8831                      ;-
8832
8833 051152 070005          TCMPP1: .WORD 070005          .SRC1 DATA TYPE = UNSIGNED PACKED, LENGTH = 5
8834 051154 051162          .WORD CMPPI1          .SRC1.DSCR
8835 051156 070005          .WORD 070005          ;SRC2 DATA TYPE = UNSIGNED PACKED, LENGTH = 5
8836 051160 051162          .WORD CMPPI1          ;SRC2.DSCR
8837
8838 051162          022      064      137  CMPPI1: .BYTE 022,064,137  ;PACKED DECIMAL STRING = 12345
8839          .EVEN
8840

```

ASHP

```

8842 .SBTTL ASHP
8843 ;*****
8844 ;
8845 ; ASHP TEST
8846 ;
8847 ;*****
8848
8849 051166 ROUTINE XASHP
8850 051166 XASHP:
051166 000004 INLINE <SCOPE> SCOPE
8851 051170 000240 INLINE <NOP> NOP
8852 051172 LET $TIMES := #20 ;ITERATION COUNT
051172 012767 000020 127204 ;GET TEST COUNT MOV #20,$TIMES
8853 051200 LET COUNT := TASH ;POINT TO FIRST TEST CASE MOV TASH,COUNT
051200 016767 013710 127262 ;INITIALIZE THE SIGN WORD MOV #TASH+2,1STCAS
8854 051206 LET TSTCAS := #TASH+2
051206 012767 065116 127330 CLR SIGNO
8855 051214 LET SIGNO := #0
051214 005067 127230
8856 ;REPEAT
8857 051220 INLINE <$5076:>
051220 $5076:
8858 051220 CALL CLRDSB ;CLEAR DECIMAL STRING BUFFERS JSR PC,CLRDSB
051220 004767 005450 CLR CC.EXP
8859 051224 LET CC.EXP := #0
051224 005067 127272
8860
8861 ;+
8862 ; GET TEST DATA FROM TABLE
8863 ;-
8864
8865 051230 LET RO := TSTCAS
051230 016700 127310 MOV TSTCAS,RO
8866 051234 LET DATA00 := (RO)+ ;SRC LENGTH MOV (RO)+,DATA00
051234 012067 127322 LET DATA01 := (RO)+ ;DST LENGTH MOV (RO)+,DATA01
8867 051240 LET DATA02 := (RO) ;SHIFT DSCR. MOV (RO),DATA02
051240 011067 127316
8869
8870 ;+
8871 ;+
8872 ; PUT THE TEST STRING INTO DSBUF1 IN SIGNED PACKED FORMAT. INCLUDE
8873 ; THE SIGN.
8874 ;
8875
8876 051250 LET RO := #DSBUF1
051250 012700 064440 MOV #DSBUF1,RO
8877 051254 LET R1 := DATA00
051254 016701 127302 MOV DATA00,R1
8878 051260 IF R1 EQ #0 THEN ;SRC.LEN = 0
051260 005701 TST R1
051262 001011 BNE $52262
8879 051264 IF SIGNO EQ #0 THEN ;SIGN IS
051264 005767 127160 TST SIGNO

```

ASHP

```

8880 051270 001003          LET (R0) :B= #14          BNE      $52263
8881 051272 112710 000014  ELSE          ;SIGN IS -   MOVB     #14,(R0)
051276 000402          ;$52263:    BR       $52264
8882 051300          LET (R0) :B= #15          MOVB     #15,(R0)
8883 051304          ENDIF          ;$52264:
8884 051304 000426  ELSE          BR       $52265
051306          ;$52262:
8885 051306          IF #BIT00 NOTSETIN R1 THEN ;SRC.LEN IS EVEN AND > 0
051306 032701 000001          BIT      #BIT00,R1
051312 001003          BNE     $52266
8886 051314          LET (R0)+ :B- #4          MOVB     #4,(R0)+
8887 051314 112720 000004          LET R1 := R1 - #1          ;DECREMENT COUNT
051320 005301          DEC     R1
8888 051322          ENDIF
8889 051322          WHILE R1 NE #1 DO          ;$52266:
051322          ;$52267:
051322 020127 000001          CMP     R1,#1
051326 001405          BEQ     $52270
8890 051330          LET (R0)+ :B= #104          ;MOVE TWO NIBBLES OF BCD
051330 112720 000104          MOVB     #104,(R0)+
8891 051334          LET R1 := R1 - #2          ;DECREMENT COUNT
051334 162701 000002          SUB     #2,R1
8892 051340          ENDDO          BR       $52267
051342          ;$52270:
8893 051342          IF SIGNO EQ #0 THEN          ;SIGN IS +
051342 005767 127102          TST     SIGNO
051346 001003          BNE     $52271
8894 051350          LET (R0) :B= #114          MOVB     #114,(R0)
8895 051354          ELSE          ;SIGN IS -   BR       $52272
051354 000402          ;$52271:
8896 051356          LET (R0) :B= #115          MOVB     #115,(R0)
8897 051356 112710 000115  ENDIF
8898 051362          ENDIF          ;$52272:
051362          ;$52265:
8899
8900          ;+
8901          ; LOAD DS.EXP AND CC.EXP WITH EXPECTED RESULTS.
8902          ;-
8903
8904          ;GET SHIFT COUNT MAGNITUDE AND PUT IT INTO TEMPO.
8905
8906 051362          LET TEMPO := DATA02
051362 016767 127200 127106  IF #BIT07 SETIN TEMPO THEN          ;NEGATIVE SHIFT COUNT
8907 051370

```

ASHP

8908	051370	032767	000200	127100		BIT	#BIT07,TEMPO
	051376	001402			LET TEMPO := NEGATE TEMPO	BEQ	\$52273
	051400					NEG	TEMPO
8909	051400	005467	127072		ENDIF		
	051404					\$52273:	
8910	051404				LET TEMPO := TEMPO CLR.BY #177600	BIC	#177600,TEMPO
	051404	042767	177600	127064			
8911							
8912					;TAKE CARE OF EXPECTED NBIT		
8913							
8914	051412				IF SIGN0 EQ #-1 AND DATA00 NE #0 THEN		
	051412	026727	127032	177777		CMP	SIGN0,-1
	051420	001006				BNE	\$52274
	051422	005767	127134			TST	DATA00
	051426	001403				BEQ	\$52274
8915	051430				LET CC.EXP := CC.EXP SET.BY #NBIT		
	051430	052767	000010	127064	ENDIF	BIS	#NBIT.CC.EXP
8916	051436					\$52274:	
	051436						
8917							
8918					;PUT THE SIGN NIBBLE INTO THE EXPECTED DESTINATION		
8919							
8920	051436				LET R1 := DATA01 ;DST.LEN		
	051436	016701	127122			MOV	DATA01,R1
8921	051442				INLINE <ASR R1>	ASR	R1
	051442	006201					
8922	051444				LET R1 := R1 + #DS.EXP	ADD	#DS.EXP,R1
	051444	062701	064606				
8923	051450				IF SIGN0 EQ #0 OR DATA00 EQ #0 THEN		
	051450	005767	126774			TST	SIGN0
	051454	001403				BEQ	\$52275
	051456	005767	127100			TST	DATA00
	051462	001003				BNE	\$52276
	051464					\$52275:	
8924	051464				LET (R1) :B= (R1) SET.BY #14 ;POSITIVE SIGN		
	051464	152711	000014			BISB	#14,(R1)
8925	051470				ELSE		
	051470	000402				BR	\$52277
	051472					\$52276:	
8926	051472				LET (R1) :B= (R1) SET.BY #15 ;NEGATIVE SIGN		
	051472	152711	000015			BISB	#15,(R1)
8927	051476				ENDIF		
	051476					\$52277:	
8928							
8929					;TAKE CARE OF ZERO LENGTH SOURCE OR DESTINATION		
8930							
8931	051476				IF DATA00 EQ #0 OR DATA01 EQ #0 THEN		
	051476	005767	127060			TST	DATA00
	051502	001403				BEQ	\$52300
	051504	005767	127054			TST	DATA01
	051510	001003				BNE	\$52301
	051512					\$52300:	
8932	051512				LET CC.EXP := CC.EXP SET.BY #ZBIT		
	051512	052767	000004	127002	ENDIF	BIS	#ZBIT.CC.EXP
8933	051520					\$52301:	
	051520						

ASHP

```

8934 051520          IF DATA00 NE #0 AND DATA01 NE #0 AND #BIT07 SETIN DATA02 THEN
      051520 005767 127036
      051524 001531
      051526 005767 127032
      051532 001526
      051534 032767 000200 127024
      051542 001522
      TST DATA00
      BEQ $52302
      TST DATA01
      BEQ $52302
      BIT #BIT07,DATA02
      BEQ $52302

8935
8936          ;TAKE CARE OF NEGATIVE SHIFT
8937
8938 051544          LET R0 := DATA00  TEMPO          ;SRC.LEN  SHIFT COUNT
      051544 016700 127012
      051550 166700 126722
      MOV DATA00,R0
      SUB TEMPO,R0
8939 051554          LET R1 := DATA01          ;DST.LEN
      051554 016701 127004
      MOV DATA01,R1
8940 051560          INLINE <ASR R1>
      051560 006201
      ASR R1
8941 051562          LET R1 := R1 + #DS.EXP
      051562 062701 064606
      ADD #DS.EXP,R1
8942 051566          LET R2 := #1
      051566 012702 000001
      MOV #1,R2
8943 051572          LET R3 := DATA01
      051572 016703 126766
      MOV DATA01,R3
8944 051576          IF R0 EQ #0 THEN
      051576 005700
      051600 001024
      TST R0
      BNE $52303
8945 051602          IF DATA02 HIS #3000 THEN
      051602 026727 126760 003000
      CMP DATA02,#3000
      BLO $52304
8946 051612          LET (R1) :B= (R1) SET.BY #20          ;ROUNDING
      051612 152711 000020
      BISB #20,(R1)
8947 051616          ELSE
      051616 000415
      BR $52305
      051620
      $52304:
8948 051620          LET CC.EXP := CC.EXP SET.BY #ZBIT
      051620 052767 000004 126674
      BIS #ZBIT,CC.EXP
8949 051626          LET CC.EXP := CC.EXP CLR.BY #NBIT
      051626 042767 000010 126666
      BIC #NBIT,CC.EXP
8950 051634          LET R5 := DATA01
      051634 016705 126724
      MOV DATA01,R5
8951 051640          INLINE <ASR R5>
      051640 006205
      ASR R5
8952 051642          LET R5 := R5 + #DS.EXP
      051642 062705 064606
      ADD #DS.EXP,R5
8953 051646          LET (R5) :B= (R5) CLR.BY #1
      051646 142715 000001
      BICB #1,(R5)
8954 051652          ENDIF
      051652
      $52305:
8955 051652          ENDIF
      051652
      $52303:
8956 051652          IF R0 LT #0 THEN
      051652 005700
      051654 002015
      TST R0
      BGE $52306
8957 051656          LET CC.EXP := CC.EXP SET.BY #ZBIT
      051656 052767 000004 126636
      BIS #ZBIT,CC.EXP
8958 051664          LET CC.EXP := CC.EXP CLR.BY #NBIT
      051664 042767 000010 126630
      BIC #NBIT,CC.EXP

```

ASMP

8959	051672			LET R5 := DATA01			
	051672	016705	126666			MOV	DATA01,R5
8960	051676			INLINE <ASR R5>			
	051676	006205				ASR	R5
8961	051700			LET R5 := R5 + #DS.EXP			
	051700	062705	064606			ADD	#DS.EXP,R5
8962	051704			LET (R5) :B= (R5) CLR.BY #1			
	051704	142715	000001			BICB	#1,(R5)
8963	051710			ENDIF			
	051710					\$52306:	
8964	051710			WHILE R2 LE R0 AND R3 HI #0 DO			
	051710					\$52307:	
	051710	020200				CMP	R2,R0
	051712	003027				BGT	\$52310
	051714	005703				TST	R3
	051716	101425				BLOS	\$52310
8965	051720			IF #BIT00 SETIN R2 THEN			
	051720	032702	000001			BIT	#BIT00,R2
	051724	001415				BEQ	\$52311
8966	051726			LET (R1) :B= (R1) SET.BY #100			
	051726	152711	000100			BISB	#100,(R1)
8967	051732			IF R2 EQ #1 AND DATA02 HIS #3000 THEN			
	051732	020227	000001			CMP	R2,#1
	051736	001006				BNE	\$52312
	051740	026727	126622 003000			CMP	DATA02,#3000
	051746	103402				BLO	\$52312
8968	051750			LET (R1) :B= (R1) SET.BY #20 ;ROUNDING			
	051750	152711	000020			BISB	#20,(R1)
8969	051754			ENDIF			
	051754					\$52312:	
8970	051754			LET R1 := R1 - #1			
	051754	005301				DEC	R1
8971	051756			ELSE			
	051756	000402				BR	\$52313
	051760					\$52311:	
8972	051760			LET (R1) :B= (R1) SET.BY #4			
	051760	152711	000004			BISB	#4,(R1)
8973	051764			ENDIF			
	051764					\$52313:	
8974	051764			LET R2 := R2 + #1			
	051764	005202				INC	R2
8975	051766			LET R3 := R3 - #1			
	051766	005303				DEC	R3
8976	051770			ENDDO			
	051770	000747				BR	\$52307
	051772					\$52310:	
8977	051772			IF R3 EQ #0 AND R2 LE R0 THEN			
	051772	005703				TST	R3
	051774	001005				BNE	\$52314
	051776	020200				CMP	R2,R0
	052000	003003				BGT	\$52314
8978	052002			LET CC.EXP := CC.EXP SET.BY #VBIT			
	052002	052767	000002 126512			BIS	#VBIT,CC.EXP
8979	052010			ENDIF			
	052010					\$52314:	
8980	052010			ENDIF			
	052010					\$52302:	

ASHP

```

8981 052010          IF DATA00 NE #0 AND DATA01 NE #0 AND #BIT07 NOTSETIN DATA02 THEN
      052010 005767 126546          TST      DATA00
      052014 001506          BEQ      $52315
      052016 005767 126542          TST      DATA01
      052022 001503          BEQ      $52315
      052024 032767 000200 126534  BIT      #BIT07,DATA02
      052032 001077          BNE      $52315

8982
8983          ;TAKE CARE OF POSITIVE SHIFT
8984
8985 052034          IF #BIT07 NOTSETIN DATA02 THEN
      052034 032767 000200 126524  BIT      #BIT07,DATA02
      052042 001073          BNE      $52316
8986 052044          LET TEMP1 := DATA00 + TEMPO          ;SRC.LEN + SHIFT COUNT
      052044 016767 126512 126426  MOV      DATA00,TEMP1
      052052 066767 126420 126420  ADD      TEMPO,TEMP1
8987 052060          IF TEMP1 HI DATA01 THEN
      052060 026767 126414 126476  CMP      TEMP1,DATA01
      052066 101403          BLOS     $52317
8988 052070          LET CC.EXP := CC.EXP SET.BY #VBIT
      052070 052767 000002 126424  BIS      #VBIT,CC.EXP
8989 052076          ENDIF
      052076          $52317:
8990 052076          IF TEMPO HIS DATA01 THEN
      052076 026767 126374 126460  CMP      TEMPO,DATA01
      052104 103407          BLO      $52320
8991 052106          LET CC.EXP := CC.EXP SET.BY #ZBIT
      052106 052767 000004 126406  BIS      #ZBIT,CC.EXP
8992 052114          LET CC.EXP := CC.EXP CLR.BY #NBIT
      052114 042767 000010 126400  BIC      #NBIT,CC.EXP
8993 052122          ELSE
      052122 000443          BR      $52321
      052124          $52320:
8994 052124          LET R4 := TEMPO + #1          ;SHIFT COUNT + 1
      052124 016704 126346          MOV      TEMPO,R4
      052130 005204          INC      R4
8995 052132          INLINE <ASR R4>
      052132 006204          ASR      R4
8996 052134          LET R1 := DATA01          ;DST.LEN
      052134 016701 126424          MOV      DATA01,R1
8997 052140          INLINE <ASR R1>
      052140 006201          ASR      R1
8998 052142          LET R1 := R1 + #DS.EXP R4
      052142 062701 064606          ADD      #DS.EXP,R1
      052146 160401          SUB      R4,R1
8999 052150          LET R0 := TEMPO          ;SHIFT COUNT
      052150 016700 126322          MOV      TEMPO,R0
9000 052154          LET R2 := DATA00          ;SRC.LEN
      052154 016702 126402          MOV      DATA00,R2
9001 052160          LET R3 := DATA01 - TEMPO  ;DST.LEN SHIFT COUNT
      052160 016703 126400          MOV      DATA01,R3
      052164 166703 126306          SUB      TEMPO,R3
9002 052170          WHILE R2 NE #0 AND R3 NE #0 JJ
      052170          $52322:
      052170 005702          TST      R2
      052172 001417          BEQ      $52323
      052174 005703          TST      R3

```



```

ASHP
9003 052176 001415
      052200
      052200 032700 000001
      052204 001004
9004 052206
      052206 152711 000100
9005 052212
      052212 005301
9006 052214
      052214 000402
      052216
9007 052216
      052216 152711 000004
9008 052222
      052222
9009 052222
      052222 005100
9010 052224
      052224 005302
9011 052226
      052226 005303
9012 052230
      052230 000757
      052232
9013 052232
      052232
9014 052232
      052232
9015 052232
      052232
9016
9017
9018
9019
9020
9021
9022
9023 052232
      052232 016700 126324
      052236 052700 060000
9024 052242
      052242 012701 064440
9025 052246
      052246 016702 126312
      052252 052702 060000
9026 052256
      052256 012703 064544
9027 052262
      052262 016704 126300
9028 052266
      052266 005005
9029 052270
      052270 076076
9030
9031
9032
9033 052272

      IF #BIT00 NOTSETIN R0 THEN
      LET (R1) :B= (R1) SET.BY #100
      LET R1 := R1 - #1
      ELSE
      LET (R1) :B= (R1) SET.BY #4
      ENDIF
      LET R0 := COMP R0
      LET R2 := R2 - #1
      LET R3 := R3 - #1
      ENDDO
      ENDIF
      ENDIF
      ENDIF

      ;+
      ; DO THE ASHP INSTRUCTION USING SIGNED PACKED (UNSIGNED
      ; PACKED WILL BE TESTED IN THE INLINE PORTION OF THE TEST).
      ; CHECK THE RESULTS AND REPORT ANY ERRORS.
      ;-

      LET R0 := DATA00 SET.BY #60000
      LET R1 := #DSBUF1
      LET R2 := DATA01 SET.BY #60000
      LET R3 := #DS.DST
      LET R4 := DATA02
      LET R5 := #0
      INLINE <ASHP>

      ;CHECK RESULTS
      LET CC := PSW

```

```

BEQ $52323
BIT #BIT00,R0
BNE $52324
BISB #100,(R1)
DEC R1
BR $52325
$52324:
BISB #4,(R1)
$52325:
;TOGGLE SHIFT COUNT
COM R0
DEC R2
DEC R3
BR $52322
$52323:
$52321:
$52316:
$52315:

```

```

MOV DATA00,R0
BIS #60000,R0
MOV #DSBUF1,R1
MOV DATA01,R2
BIS #60000,R2
MOV #DS.DST,R3
MOV DATA02,R4
CLR R5
ASHP

```

ASMP

9034	052272	016767	125500	126220	LET CC := CC CLR.BY #177760	MOV	PSW,CC
	052300					BIC	#177760,CC
9035	052306	042767	177760	126212	IF CC NE CC.EXP THEN		
	052306	026767	126206	126206		CMP	CC,CC.EXP
	052314	001402				BEQ	\$52326
9036	052316				CALL ERROR		
	052316	004767	005356			JSR	PC,ERROR
9037	052322				ENDIF		
	052322						\$52326:
9038	052322				IF R0 NE #0 OR R1 NE #0 THEN		
	052322	005700				TST	R0
	052324	001002				BNE	\$52327
	052326	005701				TST	R1
	052330	001402				BEQ	\$52330
	052332						\$52327:
9039	052332				CALL ERROR		
	052332	004767	005342			JSR	PC,ERROR
9040	052336				ENDIF		
	052336						\$52330:
9041	052336				IF R4 NE #0 OR R5 NE #0 THEN		
	052336	005704				TST	R4
	052340	001002				BNE	\$52331
	052342	005705				TST	R5
	052344	001402				BEQ	\$52332
	052346						\$52331:
9042	052346				CALL ERROR		
	052346	004767	005326			JSR	PC,ERROR
9043	052352				ENDIF		
	052352						\$52332:
9044	052352				LET TEMPO := DATA01 SET.BY #60000		
	052352	016767	126206	126116		MOV	DATA01,TEMPO
	052360	052767	060000	126110		BIS	#60000,TEMPO
9045	052366				IF R2 NE TEMPO THEN		
	052366	020267	126104			CMP	R2,TEMPO
	052372	001402				BEQ	\$52333
9046	052374				CALL ERROR		
	052374	004767	005300			JSR	PC,ERROR
9047	052400				ENDIF		
	052400						\$52333:
9048	052400				IF R3 NE #DS.DST THEN		
	052400	020327	064544			CMP	R3,#DS.DST
	052404	001402				BEQ	\$52334
9049	052406				CALL ERROR		
	052406	004767	005266			JSR	PC,ERROR
9050	052412				ENDIF		
	052412						\$52334:
9051							
9052					;SEE IF DS.DST = DS.EXP		
9053							
9054	052412				LET R0 := #40		
	052412	012700	000040			MOV	#40,R0
9055	052416				LET R1 := #DS.EXP		
	052416	012701	064606			MOV	#DS.EXP,R1
9056	052422				LET R2 := #DS.DST		
	052422	012702	064544			MOV	#DS.DST,R2
9057	052426				LET ERRFLG := #0		

ASMP

9058	052426	005067	126024			CLR	ERRFLG
	052432			WHILE RO HI #0 AND ERRFLG EQ #0 DO			
	052432				\$52335:		
	052432	005700				TST	RO
	052434	101412				BLOS	\$52336
	052436	005767	126014			TST	ERRFLG
	052442	001007				BNE	\$52336
9059	052444			IFB (R1)+ NE (R2)+ THEN			
	052444	122122				CMPB	(R1)+,(R2)+
	052446	001403				BEQ	\$52337
9060	052450			LET ERRFLG := #-1			
	052450	012767	177777	126000		MOV	#-1,ERRFLG
9061	052456			ENDIF			
	052456				\$52337:		
9062	052456			LET RO := RO - #1			
	052456	005300				DEC	RO
9063	052460			ENDDO			
	052460	000764				BR	\$52335
	052462				\$52336:		
9064	052462			IF ERRFLG NE #0 THEN			
	052462	005767	125770			TST	ERRFLG
	052466	001402				BEQ	\$52340
9065	052470			CALL ERROR		;DS.DST AND DS.EXP DO NOT MATCH	
	052470	004767	005204			JSR	PC,ERROR
9066	052474			ENDIF			
	052474				\$52340:		
9067	052474			LET SIGNO := COMP SIGNO		;REPEAT THE TEST CASE WITH A NEGATIVE	
	052474	005167	125750			COM	SIGNO
9068							
9069				; SIGN BEFORE MOVING ON TO THE NEXT TEST CASE			
9070							
9071	052500			IF SIGNO EQ #0 THEN			
	052500	005767	125744			TST	SIGNO
	052504	001005				BNE	\$52341
9072	052506			LET TSTCAS := TSTCAS + #6			
	052506	062767	000006	126030		ADD	#6,TSTCAS
9073	052514			LET COUNT := COUNT - #1			
	052514	005367	125750			DEC	COUNT
9074	052520			ENDIF			
	052520				\$52341:		
9075				;UNTIL COUNT EQ #0			
9076	052520			IF COUNT NE #0 THEN			
	052520	005767	125744			TST	COUNT
	052524	001402				BEQ	\$52342
9077	052526			INLINE <JMP \$5076>			
	052526	000167	176466			JMP	\$5076
9078	052532			ENDIF			
	052532				\$52342:		
9079							
9080							
9081							
9082				; DO ONE INLINE CASE - ASHPI. USE UNSIGNED PACKED.			
9083							
9084							
9085	052532			CALL CLRDSB			
	052532	004767	004136			JSR	PC,CLRDSB
9086	052536			LET DSBUF1 :B= #231			
							;9/9

ASHP

9087	052536	112767	000231	011674				MOVB	#231,DSBUF1
	052544				LET DSBUF1.1 :B= #231	:9/9			
9088	052544	112767	000231	011667				MOVB	#231,DSBUF1.1
	052552				LET DSBUF1.2 :B= #237	:9/17			
9089	052552	112767	000237	011662				MOVB	#237,DSBUF1.2
	052560				CALL CLRREG			JSR	PC,CLRREG
9090	052560	004767	003654						
	052564				INLINE <ASHPI>			ASHPI	
9091	052564	076176							
	052566				INLINE <.WORD TASHPI>			.WORD	TASHPI
9092	052566	052756							
	052570				INLINE <.WORD TASHPI.4>			.WORD	TASHPI.4
9093	052570	052762						.WORD	TASHPI.4
	052572				INLINE <.WORD 375>		:SHIFT COUNT = -3, RND	.WORD	DIGIT = 0
9094	052572	000375						.WORD	375
	052574				LET CC := PSW			MOV	PSW,CC
9095	052574	016767	125176	125716					
	052602				LET CC := CC CLR.BY #177760			BIC	#177760,CC
9096	052602	042767	177760	125710					
	052610				IF CC NE #0 THEN		:VBIT SHOULD BE SET	TST	CC
	052610	005767	125704					BEQ	\$52343
	052614	001402							
9097	052614				CALL ERROR			JSR	PC,ERROR
	052616	004767	005056						
9098	052616				ENDIF				
	052622								
9099	052622				IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN			\$52343:	
	052622	005700						TST	R0
	052624	001004						BNE	\$52344
	052626	005701						TST	R1
	052630	001002						BNE	\$52344
	052632	005702						TST	R2
	052634	001402						BEQ	\$52345
	052636							\$52344:	
9100	052636				CALL ERROR				
	052636	004767	005036					JSR	PC,ERROR
9101	052636				ENDIF				
	052642							\$52345:	
9102	052642				IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN				
	052642	005703						TST	R3
	052644	001004						BNE	\$52346
	052646	005704						TST	R4
	052650	001002						BNE	\$52346
	052652	005705						TST	R5
	052654	001402						BEQ	\$52347
	052656							\$52346:	
9103	052656				CALL ERROR				
	052656	004767	005016					JSR	PC,ERROR
9104	052656				ENDIF				
	052662							\$52347:	
9105	052662								
9106					:CHECK DS.DST				
9107									
9108	052662				LET R0 := #DS.DST			MOV	#DS.DST,R0
	052662	012700	064544						
9109	052662				IFB (R0). NE #11 THEN		:FIRST BYTE = 0/9	CMPB	
	052666	122027	000011						

ASHP

```

9110 052672 001402                                CALL ERROR                                BEQ      $52350
9111 052674 004767 005000                          ENDIF                                      JSR      PC.ERROR
9112 052700 001402                                IFB (R0)+ NE #237 THEN                    ;SECOND BYTE = 9/17
052700 122027 000237                                ;SECOND BYTE = 9/17                      CMPB    (R0)+,#237
052704 001402                                CALL ERROR                                BEQ      $52351
9113 052706 004767 004766                          ENDIF                                      JSR      PC.ERROR
9114 052712 001402                                LET ERRFLG := #0                          $52351:
9115 052712 005067 125540                          WHILE ERRFLG NE #0 AND R0 NE #DS.DST+37 DO CLR      ERRFLG
9116 052716 005767 125534                          $52352:
052716 001413                                TST     ERRFLG
052722 001413                                BEQ     $52353
052724 020027 064603                                CMP     R0,#DS.DST+37
052730 001410                                BEQ     $52353
9117 052732 105720                                IFB (R0)+ NE #0 THEN                      TSTB   (R0)+
052732 001405                                LET ERRFLG := #-1                        BEQ     $52354
9118 052736 012767 177777 125512                  CALL ERROR                                MOV     #-1,ERRFLG
9119 052744 004767 004750                          ENDIF                                      JSR      PC.ERROR
9120 052750 000762                                ENDDO                                     $52354:
9121 052752 000167 000010                          INLINE <JMP CIS7END>                      $52353:
9122 052752 000167 000010                                JMP     CIS7END
9123
9124 ;+
9125 ; INLINE TEST CASE
9126 ;-
9127
9128 052756 070005                                TASHPI: .WORD 070005                    ;SRC.DSCR, UNSIGNED PACKED, SRC.LEN = 5
9129 052760 064440                                .WORD DSBUF1                          ;SRC.DSCR
9130 052762 070002                                .WORD 070002                          ;DST.DSCR, UNSIGNED PACKED, DST.LEN = 2
9131 052764 064544                                .WORD DS.DST                          ;DST.DSCR
9132
9133 052766                                INLINE <CIS7END:>
052766
9134

```

CIS7END:

ASHP

```

9136 ;*****
9137 ;*****
9138 ;**
9139 ;           C I S 8   M O D U L E
9140 ;
9141 ;   THIS MODULE TESTS THE PACKED MULTIPLY INSTRUCTION: MULP.
9142 ;
9143 ;*****
9144 ;*****
9145
9146
9147 .SBTTL  CIS8 INITIALIZATION
9148 ;*****
9149 ;
9150 ;   CIS8 INITIALIZATION
9151 ;
9152 ;*****
9153
9154 ;IF SWR<5:0> ARE CLEARED OR IF SWR<4> IS SET THEN EXECUTE
9155 ; THE CIS8 MODULE; OTHERWISE, SKIP TO THE NEXT CIS MODULE.
9156
9157 052766      INLINE  <SCOPE>
          052766 000004
9158 052770      INLINE <CKSWR>
          052770 104406
9159 052772      LET TEMPO := @SWR CLR.BY #177700
          052772 017767 125430 125476
          053000 042767 177700 125470
9160 053006      IF TEMPO NE #0 THEN
          053006 005767 125464
          053012 001406
9161 053014      IF #BIT4 NOTSETIN TEMPO THEN
          053014 032767 000020 125454
          053022 001002
9162 053024      INLINE  <JMP  CIS8END>
          053024 000167 001062
9163 053030      ENDIF
          053030
9164 053030      ENDIF
          053030
9165 053030      LET CTL := #8.           ;CONTROL CHIP NUMBER
          053030 012767 000010 125510
9166 053036      LET ERRFLG := #0       ;INITIALIZE ERROR FLAG
          053036 005067 125414
9167 053042      LET PSW := #340
          053042 012767 000340 124726
9168 053050      LET ERR :- #1
          053050 012767 000001 125474
9169

```

```

SCOPE
CKSWR
MOV @SWR,TEMPO
BIC #177700,TEMPO
TST TEMPO
BEQ $52355
BIT #BIT4,TEMPO
BNE $52356
JMP CIS8END
$52356:
$52355:
MOV #8.,CTL
CLR ERRFLG
MOV #340,PSW
MOV #1,ERR

```

MULP

```

9171 .SBTTL MULP
9172 ;*****
9173 ;*
9174 ; MULP TEST
9175 ;
9176 ;*****
9177
9178 ROUTINE XMULP
9179 053056          XMULP:
9180 053056          SCOPE
000004          ;
9181 053060          NOP
000240          ;
9182 053062          LET $TIMES := #20          ;ITERATION COUNT
012767 000020 125314          ;MOV #20,$TIMES
9183 053070          LET COUNT := TAB89          ;# OF TEST CASES
016767 002166 125372          ;MOV TAB89,COUNT
9184 053076          LET TSTCAS := #TAB89+2          ;POINT TO FIRST TEST CASE
012767 055264 125440          ;MOV #TAB89-2,TSTCAS
9185
9186 ;*
9187 ; CLR R0.EXP TO R3.EXP
9188 ;-
9189          INCR R0 FROM #0 TO #6 BY #2
053104          CLR R0
053106 005000          BR $52361
053110 000402          $52362:
053110 062700 000002          ADD #2,R0
053114          $52361:
053114 020027 000006          CMP R0,#6
053120 003003          BGT $52363
9190          INCR R0
053122          CLR R0.EXP(R0)
053122 005060 000524          ENDINC
9191 053126          BR $52362
053126 000770          $52363:
053130
9192          ;REPEAT OUTER LOOP FOR TEST CASES
9193
9194          INCR R0
9195 053130          CALL CLRDSB          ;CLR DS BUFFERS
053130          $5074:
9196 053130          JSR PC,CLRDSB
004767 003540
9197
9198 ;*
9199 ; LOAD DATA0 AND CC.EXP FROM TEST CASE
9200 ;-
9201
9202 053134          LET R0 := TSTCAS
053134 016700 125404          MOV TSTCAS,R0
9203 053140          LET DATA00 := (R0)+          ;SRC1 LENGTH
053140 012067 125416          MOV (R0)+,DATA00
9204 053144          LET DATA01 := (R0)+          ;POINTER TO SRC1 DATA
053144 012067 125414          MOV (R0)+,DATA01
9205 053150          LET DATA02 := (R0)+          ;SRC2 LENGTH
053150 012067 125412          MOV (R0)+,DATA02

```

```

MULP
9206 053154          LET DATA03 := (R0)+          ;POINTER TO SRC2 DATA          MOV      (R0)+,DATA03
      053154 012067 125410
9207 053160          LET DATA04 := (R0)+          ;DEST LENGTH          MOV      (R0)+,DATA04
      053160 012067 125406
9208 053164          LET DATA05 := (R0)+          ;POINTER TO DS.EXP DATA (PRODUCT)  MOV      (R0)+,DATA05
      053164 012067 125404
9209 053170          LET CC.EXP :- (R0)+          ;CC.EXP (PRODUCT)          MOV      (R0)+,CC.EXP
      053170 012067 125326
9210
9211                ;+
9212                ;          PACK BUFFERS FOR SRC1, SRC2, & DEST
9213                ;
9214                ;
9215 053174          LET R5 := #CHRSET          ;R5 IS OPERAND STACK FOR ROUTINE  MOV      #CHRSET,R5
      053174 012705 064036
9216 053200          CALL PACK IN <#DSBUF1,DATA01,DATA00> ;SRC1          MOV      R5,-(SP)
      053200 010546          MOV      DATA00,-(R5)
      053202 016745 125354          MOV      DATA01,-(R5)
      053206 016745 125352          MOV      #DSBUF1,-(R5)
      053212 012745 064440          JSR      PC,PACK
      053216 004767 004032          MOV      (SP)+,R5
      053222 012605
9217 053224          CALL PACK IN <#DSBUF2,DATA03,DATA02> ;SRC2          MOV      R5,(SP)
      053224 010546          MOV      DATA02,-(R5)
      053226 016745 125334          MOV      DATA03,(R5)
      053232 016745 125332          MOV      #DSBUF2,(R5)
      053236 012745 064502          JSR      PC,PACK
      053242 004767 004006          MOV      (SP)+,R5
      053246 012605
9218 053250          CALL PACK IN <#DS.EXP,DATA05,DATA04> ;DEST          MOV      R5,(SP)
      053250 010546          MOV      DATA04,(R5)
      053252 016745 125314          MOV      DATA05,(R5)
      053256 016745 125312          MOV      #DS.EXP,(R5)
      053262 012745 064606          JSR      PC,PACK
      053266 004767 003762          MOV      (SP)+,R5
      053272 012605
9219
9220                ;+
9221                ;          LOAD GPR'S FROM DATAXX & ADD DATA TYPE
9222                ;
9223                ;
9224 053274          LET R0 := DATA00 SET.BY #60000          MOV      DATA00,R0
      053274 016700 125262          BIS      #60000,R0
      053300 052700 060000
9225 053304          LET R1 := #DSBUF1          MOV      #DSBUF1,R1
      053304 012701 064440
9226 053310          LET R2 := DATA02 SET.BY #60000          MOV      DATA02,R2
      053310 016702 125252          BIS      #60000,R2
      053314 052702 060000
9227 053320          LET R3 := #DSBUF2          MOV      #DSBUF2,R3
      053320 012703 064502
9228 053324          LET R4 := DATA04 SET.BY #60000          MOV      DATA04,R4
      053324 016704 125242          BIS      #60000,R4
      053330 052704 060000
9229 053334          LET R5 := #DS.DST          MOV      #DS.DST,R5
      053334 012705 064544
9230

```



MULP

```

9231      ;+
9232      ;      FINISH SET UP & DO CIS INSTRUCTION
9233      ;-
9234
9235
9236 053340      LET R4.EXP := R4
9237 053340 010467 125170      MOV      R4,R4.EXP
9238 053344      LET R5.EXP := R5
9239 053344 010567 125166      MOV      R5,R5.EXP
9240 053350      INLINE <MULP>
9241 053350 076074      MULP
9242
9243      ;+
9244      ;      CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.
9245      ;-
9246
9247 053352      LET CC := PSW
9248 053352 016767 124420 125140      MOV      PSW,CC
9249 053360      LET CC := CC CLR.BY #177760
9250 053360 042767 177760 125132      BIC      #177760,CC
9251 053366      IF CC NE CC.EXP THEN
9252 053366 026767 125126 125126      CMP      CC,CC.EXP
9253 053374 001402      BEQ      $52364
9254 053376      CALL ERROR
9255 053376 J04767 004276      JSR      PC,ERROR
9256 053402      ENDIF
9257 053402      IF R0 NE R0.EXP THEN
9258 053402 020067 125116      CMP      R0,R0.EXP
9259 053406 001402      BEQ      $52365
9260 053410      CALL ERROR
9261 053410 004767 004264      JSR      PC,ERROR
9262 053414      ENDIF
9263 053414      IF R1 NE R1.EXP THEN
9264 053414 020167 125106      CMP      R1,R1.EXP
9265 053420 001402      BEQ      $52366
9266 053422      CALL ERROR
9267 053422 004767 004252      JSR      PC,ERROR
9268 053426      ENDIF
9269 053426      IF R2 NE R2.EXP THEN
9270 053426 C20267 125076      CMP      R2,R2.EXP
9271 053432 001402      BEQ      $52367
9272 053434      CALL ERROR
9273 053434 004767 004240      JSR      PC,ERROR
9274 053440      ENDIF
9275 053440      IF R3 NE R3.EXP THEN
9276 053440 020367 125066      CMP      R3,R3.EXP
9277 053444 001402      BEQ      $52370
9278 053446      CALL ERROR
9279 053446 004767 004226      JSR      PC,ERROR
9280 053452      ENDIF
9281 053452      IF R4 NE R4.EXP THEN
9282 053452 020467 125056      CMP      R4,R4.EXP

```

MULP

```

9262 053456 001402                                CALL ERROR                                BEQ    $52371
          053460                                ENDIF                                     JSR    PC,ERROR
9263 053460 004767 004214
          053464                                ENDIF                                     $52371:
9264 053464 020567 125046                        IF R5 NE R5.EXP THEN                    CMP    R5,R5.EXP
          053464 001402                                CALL ERROR                                BEQ    $52372
9265 053470 004767 004202                                ENDIF                                     JSR    PC,ERROR
          053472 004767 004202                                ENDIF                                     $52372:
          053476
9267
9268
9269
9270
9271
9272
          ;+
          ; CHECK RESULTS VS EXPECTED RESULTS
          ;
          ;-
9273 053476                                LET R0 := #DS.DST
          053476 012700 064544                                MOV    #DS.DST,R0
9274 053502                                LET R1 := #DS.EXP
          053502 012701 064606                                MOV    #DS.EXP,R1
9275 053506                                LET ERRFLG := #0
          053506 005067 124744                                CLR    ERRFLG
9276 053512                                WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO
          053512                                $52373:
          053512 020027 064604                                CMP    R0,#DS.DST+40
          053516 001410                                JEQ    $52374
          053520 005767 124732                                TST   ERRFLG
          053524 001005                                BNE   $52374
9277 053526                                IFB (R0) NE (R1) THEN
          053526 122021                                CMPB  (R0),(R1)
          053530 001402                                BEQ   $52375
9278 053532                                CALL ERROR
          053532 004767 004142                                JSR   PC,ERROR
9279 053536                                ENDIF
          053536                                $52375:
9280 053536                                ENDDO
          053536 000765                                BR    $52373
          053540                                $52374:
9281 053540                                LET TSTCAS := TSTCAS + #22
          053540 062767 000022 124776                                ADD   #22,TSTCAS
9282 053546                                LET COUNT := COUNT - #1
          053546 005367 124716                                DEC   COUNT
9283
          ;UNTIL COUNT EQ #0
9284 053552                                IF COUNT NE #0 THEN
          053552 005767 124712                                TST   COUNT
          053556 001402                                BEQ   $52376
9285 053560                                INLINE <JMP $5074>
          053560 000167 177344                                JMP   $5074
9286 053564                                ENDIF
          053564                                $52376:
9287
9288
          ;+
9289
          ; INLINE CASE - MULPI, USING UNSIGNED PACKED DATA
9290
          ;
9291

```

## MULP

```

9292 053564          CALL CLRDSB
      053564 004767 003104          JSR      PC,CLRDSB
9293
9294          ;+
9295          ;      LOAD DATAXX AND CC.EXP FROM TEST CASE
9296          ;
9297          ;-
9298 053570          LET R0 := #TAB89+2
      053570 012700 055264          MOV      #TAB89+2,R0
9299 053574          LET DATA00 := (R0)+          ;SRC1 LENGTH
      053574 012067 124762          MOV      (R0)+,DATA00
9300 053600          LET DATA01 := (R0)+          ;POINTER TO SRC1 DATA
      053600 012067 124760          MOV      (R0)+,DATA01
9301 053604          LET DATA02 := (R0)+          ;SRC2 LENGTH
      053604 012067 124756          MOV      (R0)+,DATA02
9302 053610          LET DATA03 := (R0)+          ;POINTER TO SRC2 DATA
      053610 012067 124754          MOV      (R0)+,DATA03
9303 053614          LET DATA04 := (R0)+          ;DEST LENGTH
      053614 012067 124752          MOV      (R0)+,DATA04
9304 053620          LET DATA05 := (R0)+          ;POINTER TO DS.EXP DATA (PRODUCT)
      053620 012067 124750          MOV      (R0)+,DATA05
9305
9306          ;+
9307          ;      PACK BUFFERS FOR SRC1,SRC2, & DEST
9308          ;
9309          ;-
9310 053624          LET R5 := #CHRSET          ;R5 IS OPERAND STACK FOR ROUTINE
      053624 012705 064036          MOV      #CHRSET,R5
9311 053630          CALL PACK IN <#DSBUF1,DATA01,DATA00> ;SRC1
      053630 010546          MOV      R5, (SP)
      053632 016745 124724          MOV      DATA00, (R5)
      053636 016745 124722          MOV      DATA01, (R5)
      053642 012745 064440          MOV      #DSBUF1, (R5)
      053646 004767 003402          JSR      PC,PACK
      053652 012605          MOV      (SP)+,R5
9312 053654          CALL PACK IN <#DSBUF2,DATA03,DATA02> ;SRC2
      053654 010546          MOV      R5, (SP)
      053656 016745 124704          MOV      DATA02, -(R5)
      053662 016745 124702          MOV      DATA03, -(R5)
      053666 012745 064502          MOV      #DSBUF2, (R5)
      053672 004767 003356          JSR      PC,PACK
      053676 012605          MOV      (SP)+,R5
9313 053700          CALL PACK IN <#DS.EXP,DATA05,DATA04> ;DEST
      053700 010546          MOV      R5, -(SP)
      053702 016745 124664          MOV      DATA04, (R5)
      053706 016745 124662          MOV      DATA05, -(R5)
      053712 012745 064606          MOV      #DS.EXP, (R5)
      053716 004767 003332          JSR      PC,PACK
      053722 012605          MOV      (SP)+,R5
9314 053724          CALL CLRREG          ;ENSURE GPR'S 0-5 ARE ZERO
      053724 004767 002510          JSR      PC,CLRREG
9315
9316 053730          INLINE <MULPI>
      053730 076174          MULPI
9317 053732          INLINE <.WORD TMULP1>
      053732 056064          .WORD TMULPI
9318 053734          INLINE <.WORD TMULPI+4>

```

MULP

```

053734 056070 .WORD TMULPI+4
9319 053736 INLINE <.WORD TMULPI+10>
053736 056074 .WORD TMULPI+10
9320 053740 LET CC := PSW MOV PSW,CC
053740 016767 124032 124552 LET CC := CC CLR.BY #177760 BIC #177760,CC
9321 053746 LET CC := CC CLR.BY #177760
053746 042767 177760 124544 IF CC NE #0 THEN TST CC
9322 053754 IF CC NE #0 THEN BEQ $52377
053754 005767 124540 CALL ERROR JSR PC,ERROR
053760 001402 ENDIF
9323 053762 CALL ERROR JSR PC,ERROR
053762 004767 003712 ENDIF
9324 053766 $52377:
053766
9325
9326 ;+
9327 ; ENSURE GPR'S STILL ZERO
9328 ;
9329 ;-
9330 053766 IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN
053766 005700 TST R0
053770 001004 BNE $52400
053772 005701 TST R1
053774 001002 BNE $52400
053776 005702 TST R2
054000 001402 BEQ $52401
054002 $52400:
9331 054002 CALL ERROR JSR PC,ERROR
054002 004767 003672 ENDIF
9332 054006 $52401:
054006 IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
9333 054006 TST R3
054010 001004 BNE $52402
054012 005704 TST R4
054014 001002 BNE $52402
054016 005705 TST R5
054020 001402 BEQ $52403
054022 $52402:
9334 054022 CALL ERROR JSR PC,ERROR
054022 004767 003652 ENDIF
9335 054026 $52403:
054026
9336
9337 ;+
9338 ; CHECK RESULTS
9339 ;
9340
9341 054026 LET R0 := #DS.DST MOV #DS.DST,R0
054026 012700 064544 LET R1 := #DS.EXP ;EXPECTED DATA MOV #DS.EXP,R1
9342 054032 LET R2 := TMULPI+10 CLR.BY #70000 ;GET STRING LENGTH MOV TMULPI+10,R2
054032 012701 064606 BIC #70000,R2
9343 054036 LET R2 := TMULPI+10 CLR.BY #70000 ;GET STRING LENGTH MOV TMULPI+10,R2
054036 016702 002032 BIC #70000,R2
054042 042702 070000
9344 054046 INLINE <ASR R2> ;LENGTH/2 (PACKING FACTOR) ASR R2
054046 006202

```



MULP

9357  
9358  
9359  
9360  
9361  
9362  
9363  
9364  
9365  
9366  
9367  
9368  
9369  
9370  
9371  
9372  
9373  
9374  
9375  
9376  
9377  
9378  
9379  
9380  
9381  
9382  
9383  
9384  
9385  
9386  
9387  
9388  
9389  
9390

```
*****  
*****  
; **  
;           C I S 9   M O D U L E  
; :  
; THIS MODULE TESTS THE PACKED DIVIDE INSTRUCTION: DIVP.  
; --  
; *****  
; *****
```

```
.SBTTL  CIS9 INITIALIZATION  
; *****  
; :  
; CIS9 INITIALIZATION  
; :  
; *****
```

```
; IF SWR<5:0> ARE CLEARED OR IF SWR<5> IS SET THEN EXECUTE  
; THE CIS9 MODULE; OTHERWISE, SKIP TO END OF PASS.
```

```
INLINE <SCOPE>
```

```
SCOPE
```

```
INLINE <CKSWR>
```

```
CKSWR
```

```
LET TEMPO := @SWR CLR.BY #177700
```

```
MOV @SWR,TEMPO  
BIC #177700,TEMPO
```

```
IF TEMPO NE #0 THEN
```

```
TST TEMPO  
BEQ $52407
```

```
IF #BITS NOTSETIN TEMPO THEN
```

```
BIT #BITS,TEMPO  
BNE $52410
```

```
INLINE <JMP CIS9END>
```

```
JMP CIS9END
```

```
ENDIF
```

```
$52410:
```

```
ENDIF
```

```
$52407:
```

```
LET CTL := #9. ;CONTROL CHIP NUMBER
```

```
MOV #9.,CTL
```

```
LET ERRFLG := #0 ;INITIALIZE ERROR FLAG
```

```
CLR ERRFLG
```

```
LET PSW := #340
```

```
MOV #340,PSW
```

```
LET ERR := #1
```

```
MOV #1,ERR
```

DIVP

```

9392          .SBTTL  DIVP
9393          ;*****
9394          ;+
9395          ;      DIVP TEST
9396          ;-
9397          ;*****
9398
9399 054202    ROUTINE XDIVP
054202
9400 054202    INLINE <SCOPE>                                XDIVP:
054202 000004    SCOPE
9401 054204    INLINE <NOP>                                  NOP
054204 000240    LET $TIMES := #20                          ;ITERATION COUNT
9402 054206    012767 000020 124170                       ;MOV #20,$TIMES
054206 012767 000020 124170                               ;LET COUNT := TAB89 ;# OF TEST CASES
9403 054214    016767 001042 124246                       ;LET TSTCAS := #TAB89+2 ;POINT TO FIRST TEST CASE
054214 016767 001042 124246                               ;MOV TAB89,COUNT
9404 054222    012767 055264 124314                       ;MOV #TAB89+2,TSTCAS
054222 012767 055264 124314
9405
9406          ;+
9407          ;      CLR R0.EXP TO R3.EXP
9408          ;-
9409
9410 054230    INCR R0 FROM #0 TO #6 BY #2
054230 005000
054232 000402
054234 062700 000002
054240 020027 000006
054244 003003
9411 054246    INLINE <CLR R0.EXP(R0)>
054246 005060 000524
9412 054252    ENDINC
054252 000770
054254
9413          ;REPEAT OUTER LOOP FOR TEST CASES
9414 054254    INLINE <#5075:>
054254
9415 054254    CALL CLRDSB                                ;CLR DS BUFFERS
054254 004767 002414
9416
9417          ;+
9418          ;      LOAD DATAXX AND CC.EXP FROM TEST CASE
9419          ;-
9420
9421 054260    LET R0 := TSTCAS
054260 016700 124260
9422 054264    LET DATA00 := (R0)+                      ;SRC1 LENGTH
054264 012067 124272
9423 054270    LET DATA01 := (R0)+                      ;POINTER TO SRC1 DATA
054270 012067 124270
9424 054274    LET DATA02 := (R0)+                      ;SRC2 LENGTH
054274 012067 124266
9425 054300    LET DATA03 := (R0)+                      ;POINTER TO SRC2 DATA
054300 012067 124264

```

DIVP

```

9426 054304          LET DATA04 := (R0)•          ;DEST LENGTH          MOV      (R0)•,DATA04
      054304 012067 124262
9427 054310          LET DATA05 := (R0)•          ;POINTER TO DS.EXP DATA (PRODUCT)  MOV      (R0)•,DATA05
      054310 012067 124260
9428 054314          LET CC.EXP := (R0)•          ;CC.EXP (PRODUCT)          MOV      (R0)•,CC.EXP
      054314 012067 124202
9429 054320          LET DATA05 := (R0)•          ;POINTER TO DS.EXP DATA (QUOTIENT)  MOV      (R0)•,DATA05
      054320 012067 124250
9430 054324          LET CC.EXP := (R0)•          ;CC.EXP (QUOTIENT)        MOV      (R0)•,CC.EXP
      054324 012067 124172
9431
9432                ;+
9433                ;      PACK BUFFERS FOR SRC1, SRC2, & DEST
9434                ;-
9435
9436 054330          LET R5 := #CHRSET          ;R5 IS OPERAND STACK FOR ROUTINE
      054330 012705 064036          MOV      #CHRSET, R5
9437 054334          CALL PACK IN <#DSBUF1, DATA01, DATA00> ;SRC1
      054334 010546          MOV      R5, (SP)
      054336 016745 124220          MOV      DATA00, (R5)
      054342 016745 124216          MOV      DATA01, (R5)
      054346 012745 064440          MOV      #DSBUF1, -(R5)
      054352 004767 002676          JSR      PC, PACK
      054356 012605          MOV      (SP)•, R5
9438 054360          CALL PACK IN <#DSBUF2, DATA03, DATA02> ;SRC2
      054360 010546          MOV      R5, (SP)
      054362 016745 124200          MOV      DATA02, (R5)
      054366 016745 124176          MOV      DATA03, (R5)
      054372 012745 064502          MOV      #DSBUF2, (R5)
      054376 004767 002652          JSR      PC, PACK
      054402 012605          MOV      (SP)•, R5
9439 054404          CALL PACK IN <#DS.EXP, DATA05, DATA04> ;DEST
      054404 010546          MOV      R5, (SP)
      054406 016745 124160          MOV      DATA04, (R5)
      054412 016745 124156          MOV      DATA05, (R5)
      054416 012745 064606          MOV      #DS.EXP, (R5)
      054422 004767 002626          JSR      PC, PACK
      054426 012605          MOV      (SP)•, R5
9440
9441                ;+
9442                ;      LOAD GPR'S FROM DATAXX & ADD DATA TYPE
9443                ;-
9444
9445 054430          LET R0 := DATA00 SET.BY #60000
      054430 016700 124126          MOV      DATA00, R0
      054434 052700 060000          BIS      #60000, R0
9446 054440          LET R1 := #DSBUF1
      054440 012701 064440          MOV      #DSBUF1, R1
9447 054444          LET R2 := DATA02 SET.BY #60000
      054444 016702 124116          MOV      DATA02, R2
      054450 052702 060000          BIS      #60000, R2
9448 054454          LET R3 := #DSBUF2
      054454 012703 064502          MOV      #DSBUF2, R3
9449 054460          LET R4 := DATA04 SET.BY #60000
      054460 016704 124106          MOV      DATA04, R4
      054464 052704 060000          BIS      #60000, R4
9450 054470          LET R5 := #DS.DST

```



DIVP

9451	054470	012705	064544			MOV	#DS.DST.R5
9452				;			
9453				;	FINISH SET UP & DO CIS INSTRUCTION		
9454				;-			
9455							
9456							
9457	054474				LET R4.EXP := R4		
9458	054474	010467	124034			MOV	R4,R4.EXP
9459	054500				LET R5.EXP := R5		
	054500	010567	124032			MOV	R5,R5.EXP
	054504				INLINE <DIVP>		
	054504	076075				DIVP	
9460							
9461				;			
9462				;	CHECK CONDITION CODES AND GPR'S FOR CORRECT RESULTS.		
9463				;-			
9464							
9465	054506				LET CC := PSW		
	054506	016767	123264	124004		MOV	PSW,CC
9466	054514				LET CC := CC CLR.BY #177760		
	054514	042767	177760	123776		BIC	#177760,CC
9467	054522				IF CC NE CC.EXP THEN		
	054522	026767	123772	123772		CMP	CC,CC EXP
	054530	001402				BEQ	\$52416
9468	054532				CALL ERROR		
	054532	004767	003142			JSR	PC,ERROR
9469	054536				ENDIF		
	054536					\$52416:	
9470	054536				IF R0 NE R0.EXP THEN		
	054536	020067	123762			CMP	R0,R0.EXP
	054542	001402				BEQ	\$52417
9471	054544				CALL ERROR		
	054544	004767	003130			JSR	PC,ERROR
9472	054550				ENDIF		
	054550					\$52417:	
9473	054550				IF R1 NE R1.EXP THEN		
	054550	020167	123752			CMP	R1,R1.EXP
	054554	001402				BEQ	\$52420
9474	054556				CALL ERROR		
	054556	004767	003116			JSR	PC,ERROR
9475	054562				ENDIF		
	054562					\$52420:	
9476	054562				IF R2 NE R2.EXP THEN		
	054562	020267	123742			CMP	R2,R2.EXP
	054566	001402				BEQ	\$52421
9477	054570				CALL ERROR		
	054570	004767	003104			JSR	PC,ERROR
9478	054574				ENDIF		
	054574					\$52421:	
9479	054574				IF R3 NE R3.EXP THEN		
	054574	020367	123732			CMP	R3,R3.EXP
	054600	001402				BEQ	\$52422
9480	054602				CALL ERROR		
	054602	004767	003072			JSR	PC,ERROR
9481	054606				ENDIF		
	054606					\$52422:	

DIVP

```

9482 054606          IF R4 NE R4.EXP THEN
      054606 020467 123722
      054612 001402
9483 054614          CALL ERROR
      054614 004767 003060
9484 054620          ENDIF
      054620
9485 054620          IF R5 NE R5.EXP THEN
      054620 020567 123712
      054624 001402
9486 054626          CALL ERROR
      054626 004767 003046
9487 054632          ENDIF
      054632
9488
9489
9490
9491
9492
9493
9494 054632          LET R0 := #DS.DST
      054632 012700 064544
9495 054636          LET R1 := #DS.EXP
      054636 012701 064606
9496 054642          LET ERRFLG := #0
      054642 005067 123610
9497 054646          WHILE R0 NE #DS.DST+40 AND ERRFLG EQ #0 DO
      054646
      054646 020027 064604
      054652 001410
      054654 005767 123576
      054660 001005
9498 054662          IFB (R0)+ NE (R1)+ THEN
      054662 122021
      054664 001402
9499 054666          CALL ERROR
      054666 004767 003006
9500 054672          ENDIF
      054672
9501 054672          ENDDO
      054672 000765
      054674
9502 054674          LET TSTCAS := TSTCAS + #22
      054674 062767 000022 123642
9503 054702          LET COUNT := COUNT - #1
      054702 005367 123562
9504
9505 054706          ;UNTIL COUNT EQ #0
      054706 005767 123556
      054712 001402
9506 054714          INLINE <JMP $5075>
      054714 000167 177334
9507 054720          ENDIF
      054720
9508
9509
9510

```

```

CMP      R4,R4.EXP
BEQ      $52423
JSR      PC.ERROR
$52423:
CMP      R5,R5.EXP
BEQ      $52424
JSR      PC.ERROR
$52424:
MOV      #DS.DST,R0
MOV      #DS.EXP,R1
CLR      ERRFLG
$52425:
CMP      R0,#DS.DST+40
BEQ      $52426
TST      ERRFLG
BNE      $52426
CMPB     (R0)+,(R1)+
BEQ      $52427
JSR      PC.ERROR
$52427:
BR       $52425
$52426:
ADD      #22,TSTCAS
DEC      COUNT
TST      COUNT
BEQ      $52430
JMP      $5075
$52430:

```

;;  
; INLINE CASE DIVPI, USING UNSIGNED PACKED DATA

DIVP

```

9511      ;
9512      ;
9513 054720      CALL CLRDSB
054720 004767 001750      JSR      PC,CLRDSB
9514
9515      ;*
9516      ;      LOAD DATAX AND CC.EXP FROM TEST CASE
9517      ;
9518
9519 054724      LET R0 := #TAB89+2
054724 012700 055264      MOV      #TAB89+2,R0
9520 054730      LET DATA00 := (R0)+      ;SRC1 LENGTH
054730 012067 123626      MOV      (R0)+,DATA00
9521 054734      LET DATA01 := (R0)+      ;POINTER TO SRC1 DATA
054734 012067 123624      MOV      (R0)+,DATA01
9522 054740      LET DATA02 := (R0)+      ;SRC2 LENGTH
054740 012067 123622      MOV      (R0)+,DATA02
9523 054744      LET DATA03 := (R0)+      ;POINTER TO SRC2 DATA
054744 012067 123620      MOV      (R0)+,DATA03
9524 054750      LET DATA04 := (R0)+      ;DEST LENGTH
054750 012067 123616      MOV      (R0)+,DATA04
9525 054754      LET DATA05 := (R0)+      ;POINTER TO DS.EXP DATA (PRODUCT)
054754 012067 123614      MOV      (R0)+,DATA05
9526 054760      LET CC.EXP := (R0)+
054760 012067 123536      MOV      (R0)+,CC.EXP
9527 054764      LET DATA05 := (R0)+      ;POINTER TO DS.EXP DATA (QUOTIENT)
054764 012067 123604      MOV      (R0)+,DATA05
9528
9529      ;*
9530      ;      PACK BUFFERS FOR SRC1, SRC2, & DEST
9531      ;
9532
9533 054770      LET R5 := #CHRSET      ;R5 IS OPERAND STACK FOR ROUTINE
054770 012705 064036      MOV      #CHRSET,R5
9534 054774      CALL PACK IN <#DSBUF1,DATA01,DATA00> ;SRC1
054774 010546      MOV      R5,-(SP)
054776 016745 123560      MOV      DATA00,-(R5)
055002 016745 123556      MOV      DATA01,-(R5)
055006 012745 064440      MOV      #DSBUF1,-(R5)
055012 004767 002236      JSR      PC,PACK
055016 012605      MOV      (SP)+,R5
9535 055020      CALL PACK IN <#DSBUF2,DATA03,DATA02> ;SRC2
055020 010546      MOV      R5,(SP)
055022 016745 123540      MOV      DATA02,-(R5)
055026 016745 123536      MOV      DATA03,(R5)
055032 012745 064502      MOV      #DSBUF2,-(R5)
055036 004767 002212      JSR      PC,PACK
055042 012605      MOV      (SP)+,R5
9536 055044      CALL PACK IN <#DS.EXP,DATA05,DATA04> ;DEST
055044 010546      MOV      R5,-(SP)
055046 016745 123520      MOV      DATA04,(R5)
055052 016745 123516      MOV      DATA05,-(R5)
055056 012745 064606      MOV      #DS.EXP,-(R5)
055062 004767 002166      JSR      PC,PACK
055066 012605      MOV      (SP)+,R5
9537 055070      CALL CLRREG      ;ENSURE GPR S O S ARE ZERO
055070 004767 001344      JSR      PC,CLRREG
    
```

DIVP

```

9538
9539 055074          INLINE <DIVPI>
          055074 076175
9540 055076          INLINE <.WORD TMULPI>
          055076 056064
9541 055100          INLINE <.WORD TMULPI*4>
          055100 056070
9542 055102          INLINE <.WORD TMULPI*10>
          055102 056074
9543 055104          LET CC := PSW
          055104 016767 122666 123406
9544 055112          LET CC := CC CLR.BY #177760
          055112 042767 177760 123400
9545 055120          IF CC NE #0 THEN
          055120 005767 123374
          055124 001402
9546 055126          CALL ERROR
          055126 004767 002546
9547 055132          ENDIF
          055132
9548
9549
9550
9551
9552
9553 055132          IF R0 NE #0 OR R1 NE #0 OR R2 NE #0 THEN
          055132 005700
          055134 001004
          055136 005701
          055140 001002
          055142 005702
          055144 001402
          055146
9554 055146          CALL ERROR
          055146 004767 002526
9555 055152          ENDIF
          055152
9556 055152          IF R3 NE #0 OR R4 NE #0 OR R5 NE #0 THEN
          055152 005703
          055154 001004
          055156 005704
          055160 001002
          055162 005705
          055164 001402
          055166
9557 055166          CALL ERROR
          055166 004767 002506
9558 055172          ENDIF
          055172
9559
9560
9561
9562
9563
9564 055172          LET R0 := #DS.DST
          055172 012700 064544
9565 055176          LET R1 := #DS.EXP
          ;EXPECTED DATA

```

```

DIVPI
.WORD TMULPI
.WORD TMULPI*4
.WORD TMULPI*10
MOV PSW,CC
BIC #177760,CC
TST CC
BEQ $52431
JSR PC,ERROR
$52431:
TST R0
BNE $52432
TST R1
BNE $52432
TST R2
BEQ $52433
$52432:
JSR PC,ERROR
$52433:
TST R3
BNE $52434
TST R4
BNE $52434
TST R5
BEQ $52435
$52434:
JSR PC,ERROR
$52435:
MOV #DS.DST,R0

```

DIVP

9566	055176	012701	064606	LET R2 := TMULPI*10 CLR.BY #70000	;GET STRING LENGTH	MOV	#DS.EXP,R1
	055202	016702	000666			MOV	TMULPI*10,R2
	055206	042702	070000			BIC	#70000,R2
9567	055212			INLINE <ASR R2>	;LENGTH/2 (PACKING FACTOR)	ASR	R2
	055212	006202					
9568	055214			LET R2 := R2 * #DS.EXP	;R2 POINTS TO MSD/SIGN	ADD	#DS.EXP,R2
	055214	062702	064606				
9569	055220			LET (R2) :B= (R2) SET.BY #17 ;MAKE IT UNSIGNED		BISB	#17,(R2)
	055220	152712	000017				
9570	055224			LET ERRFLG := #0		CLR	ERRFLG
	055224	005067	123226				
9571	055230			WHILE R0 NE #DS.DST*37 AND ERRFLG EQ #0 DO			
	055230					\$52436:	
	055230	020027	064603			CMPE	R0,#DS.DST*37
	055234	001410				BEQ	\$52437
	055236	005767	123214			TST	ERRFLG
	055242	001005				BNE	\$52437
9572	055244			IFB (R0) NE (R1) THEN			
	055244	122021				CMPE	(R0), (R1)
	055246	001402				BEQ	\$52440
9573	055250			CALL ERROR		JSR	PC,ERROR
	055250	004767	002424				
9574	055254			ENDIF			
	055254					\$52440:	
9575	055254			ENDDO			
	055254	000765				BR	\$52436
	055256					\$52437:	
9576							
9577	055256			INLINE <JMP CIS9END>		JMP	CIS9END
	055256	000167	000616				
9578							

TEST CASES FOR CIS8 & CIS9

			.SBTTL	TEST CASES FOR CIS8 & CIS9	
9580					
9581					
9582	055262	000007	TAB89:	.WORD 7	:# OF TEST CASES
9583					
9584	055264	000017		.WORD 15.	;SRC1 LENGTH
9585	055266	055462		.WORD MUL1	;SRC1 PTR
9586	055270	000021		.WORD 17.	;SRC2 LENGTH
9587	055272	055504		.WORD MUL1A	;SRC2 PTR
9588	055274	000037		.WORD 31.	;DEST LENGTH
9589	055276	055530		.WORD MUL1B	;DEST PTR (PRODUCT)
9590	055300	000000		.WORD 0	;CC.EXP (PRODUCT)
9591	055302	055572		.WORD MUL1C	;DEST PTR (QUOTIENT)
9592	055304	000000		.WORD 0	;CC.EXP (QUOTIENT)
9593					
9594	055306	000004		.WORD 4	
9595	055310	055634		.WORD MUL2	
9596	055312	000007		.WORD 7	
9597	055314	055642		.WORD MUL2A	
9598	055316	000013		.WORD 11.	
9599	055320	055654		.WORD MUL2B	
9600	055322	000010		.WORD 10	
9601	055324	055672		.WORD MUL2C	
9602	055326	000010		.WORD 10	
9603					
9604	055330	000005		.WORD 5	
9605	055332	055710		.WORD MUL3	
9606	055334	000002		.WORD 2	
9607	055336	055720		.WORD MUL3A	
9608	055340	000006		.WORD 6	
9609	055342	055724		.WORD MUL3B	
9610	055344	000000		.WORD 0	
9611	055346	055734		.WORD MUL3C	
9612	055350	000004		.WORD 4	
9613					
9614	055352	000005		.WORD 5	
9615	055354	055744		.WORD MUL4	
9616	055356	000001		.WORD 1	
9617	055360	055754		.WORD MUL4A	
9618	055362	000001		.WORD 1	
9619	055364	055762		.WORD MUL4B	
9620	055366	000004		.WORD 4	
9621	055370	055770		.WORD MUL4C	
9622	055372	000004		.WORD 4	
9623					
9624	055374	000002		.WORD 2	
9625	055376	055776		.WORD MUL5	
9626	055400	000005		.WORD 5	
9627	055402	056002		.WORD MUL5A	
9628	055404	000000		.WORD 0	
9629	055406	056012		.WORD MUL5B	
9630	055410	000006		.WORD 6	
9631	055412	056016		.WORD MUL5C	
9632	055414	000006		.WORD 6	
9633					
9634	055416	000005		.WORD 5	
9635	055420	055744		.WORD MUL4	
9636	055422	000001		.WORD 1	

TEST CASES FOR CIS8 & CIS9

9637	055424	055754				.WORD	MUL4A
9638	055426	000004				.WORD	4
9639	055430	055762				.WORD	MUL4B
9640	055432	000004				.WORD	4
9641	055434	055770				.WORD	MUL4C
9642	055436	000004				.WORD	4
9643							
9644	055440	000004				.WORD	4
9645	055442	056022				.WORD	MUL7
9646	055444	000005				.WORD	5
9647	055446	056030				.WORD	MUL7A
9648	055450	000010				.WORD	8.
9649	055452	056040				.WORD	MUL7B
9650	055454	000000				.WORD	0
9651	055456	056052				.WORD	MUL7C
9652	055460	000000				.WORD	0
9653							
9654	055462	000000			MUL1:	.WORD	0
9655	055464	004	004	004		.BYTE	4,4,4,4,4,4,4,4,4,4,4,4,4,4,4
	055467	004	004	004			
	055472	004	004	004			
	055475	004	004	004			
	055500	004	004	004			
9656						.EVEN	
9657							
9658	055504	000000			MUL1A:	.WORD	0
9659	055506	001	007	004		.BYTE	1,7,4,3,2,1,6,9.,5,3,4,7,9.,4,1,6,2
	055511	003	002	001			
	055514	006	011	005			
	055517	003	004	007			
	055522	011	004	001			
	055525	006	002				
9660						.EVEN	
9661							
9662	055530	000000			MUL1B:	.WORD	0
9663	055532	007	007	004		.BYTE	7,7,4,7,6,3,0,9.,0,4,3,5,2,9.,5,3
	055535	007	006	003			
	055540	000	011	000			
	055543	004	003	005			
	055546	002	011	005			
	055551	003					
9664	055552	001	004	001		.BYTE	1,4,1,2,5,7,9.,8.,4,5,3,5,9.,2,8.
	055555	002	005	007			
	055560	011	010	004			
	055563	005	003	005			
	055566	011	002	010			
9665						.EVEN	
9666							
9667	055572	000000			MUL1C:	.WORD	0
9668	055574	000	000	000		.BYTE	0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
	055577	000	000	000			
	055602	000	000	000			
	055605	000	000	000			
	055610	000	000	000			
	055613	000					
9669	055614	000	000	000		.BYTE	0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,9.
	055617	000	000	000			

TEST CASES FOR CIS8 & CIS9

055622	000	000	000			
055625	000	000	000			
055630	000	003	011			
9670						.EVEN
9671						
9672	055634	177777			MUL2:	.WORD -1
9673	055636	002	007	004		.BYTE 2,7,4,1
	055641	001				
9674						.EVEN
9675						
9676	055642	000000			MUL2A:	.WORD 0
9677	055644	010	004	006		.BYTE 8..4.6,2,9.,7,3
	055647	002	011	007		
	055652	003				
9678						.EVEN
9679	055654	177777			MUL2B:	.WORD -1
9680	055656	002	003	001		.BYTE 2,3,1,9.,7,0,0,8.,9.,9.,3
	055661	011	007	000		
	055664	000	010	011		
	055667	011	003			
9681						.EVEN
9682						
9683	055672	177777			MUL2C:	.WORD -1
9684	055674	000	000	000		.BYTE 0,0,0,0,0,0,0,3,0,8.,7
	055677	000	000	000		
	055702	000	003	000		
	055705	010	007			
9685						.EVEN
9686						
9687	055710	177777			MUL3:	.WORD -1
9688	055712	004	001	007		.BYTE 4,1,7,3,9.
	055715	003	011			
9689						.EVEN
9690						
9691	055720	177777			MUL3A:	.WORD -1
9692	055722	001	001			.BYTE 1,1
9693						.EVEN
9694						
9695	055724	000000			MUL3B:	.WORD 0
9696	055726	004	005	011		.BYTE 4,5,9.,1,2,9.
	055731	001	002	011		
9697						.EVEN
9698						
9 39	055734	000000			MUL3C:	.WORD 0
9700	055736	000	000	000		.BYTE 0,0,0,0,0,0
	055741	000	000	000		
9701						.EVEN
9702						
9703	055744	000000			MUL4:	.WORD 0
9704	055746	001	007	003		.BYTE 1,7,3,5,4
	055751	005	004			
9705						.EVEN
9706						
9707	055754	000000			MUL4A:	.WORD 0
9708	055756	000	000	000		.BYTE 0,0,0,0
	055761	000				
9709						.EVEN



TEST CASES FOR CIS8 & CIS9

```

9710
9711 055762 000000
9712 055764 000 000 000 MUL4B: .WORD 0
      055767 000 .BYTE 0,0,0,0
9713 .EVEN
9714
9715 055770 000000
9716 055772 000 000 000 MUL4C: .WORD 0
      055775 000 .BYTE 0,0,0,0
9717 .EVEN
9718
9719 055776 000000
9720 056000 001 007 MUL5: .WORD 0
9721 .BYTE 1,7
9722 .EVEN
9723 056002 000000
9724 056004 005 002 000 MUL5A: .WORD 0
      056007 000 .BYTE 5,2,0,0,0
9725 .EVEN
9726
9727 056012 000000
9728 056014 000 MUL5B: .WORD 0
9729 .BYTE 0
9730 .EVEN
9731 056016 000000
9732 056020 000 MUL5C: .WORD 0
9733 .BYTE 0
9734 .EVEN
9735 ;MUL6 USES THE MUL4 STRINGS
9736
9737 056022 000000
9738 056024 001 000 000 MUL7: .WORD 0
      056027 001 .BYTE 1,0,0,1
9739 .EVEN
9740 056030 000000
9741 056032 001 000 000 MUL7A: .WORD 0
      056035 000 .BYTE 1,0,0,0,0
9742 .EVEN
9743 056040 000000
9744 056042 001 000 000 MUL7B: .WORD 0
      056045 001 000 000 .BYTE 1,0,0,1,0,0,0,0
      056050 000 000
9745 .EVEN
9746 056052 000000
9747 056054 000 000 000 MUL7C: .WORD 0
      056057 000 000 000 .BYTE 0,0,0,0,0,0,0,9.
      056062 000 011
9748 .EVEN
9749
9750 ;+
9751 ; INLINE TEST CASE FOR MULP & DIVP
9752 ;
9753
9754 056064 070017 TMULPI: .WORD 070017
9755 056066 064440 .WORD DSBUF1
9756 056070 070021 .WORD 070021
9757 056072 064502 .WORD DSBUF2

```

TEST CASES FOR CIS8 & CIS9

```

9758 056074 070037 .WORD 070037
9759 056076 064544 .WORD DS.DST
9760
9761
9762 056100 CIS9END:
9763
9764
9765 ;+
9766 ; BEFORE GOING TO $EOP DETERMINE IF TESTING WITH AND WITHOUT
9767 ; MEMORY MANAGEMENT ENABLED HAS OCCURRED IF APPLICABLE.
9768 ; IF MEMORY MANAGEMENT DISABLED THEN CLEAR TSTNM AND ITERATION
9769 ; COUNT AND GO TEST WITH MEMORY MANAGEMENT ENABLED.
9770 ;-
9771
9772 056100 IF @#MMCYL EQ #0 AND #BIT07 NOTSETIN @SWR THEN
          056100 005737 000446 TST @#MMCYL
          056104 001012 BNE $52441
          056106 032777 000200 122312 BIT #BIT07,@SWR
          056114 001006 BNE $52441
9773 056116 LET $TSTNM := #0 CLR $TSTNM
9774 056122 LET $TIMES := #0 CLR $TIMES
9775 056126 INLINE <JMP BEGIN> JMP BEGIN
          056126 000167 123506
9776 056132 ENDIF
          056132 $52441:
9777
9778 ;.ENABL ABS
9779 ;:*****
9780 ;.SBTTL END OF PASS ROUTINE
9781 ;:*****
9782 ;*INCREMENT THE PASS NUMBER ($PASS)
9783 ;*TYPE "END PASS #XXXXX" (WHERE XXXXX IS A DECIMAL NUMBER)
9784 ;*IF THERES A MONITOR GO TO IT
9785 ;*IF THERE ISN'T JUMP TO BEGIN
9786
9787 $EOP:
9788 056132 000004 SCOPE
9789 056134 005067 122242 CLR $TSTNM ;;ZERO THE TEST NUMBER
9790 056140 005067 122240 CLR $TIMES
9791 056144 005267 122470 INC $PASS ;;INCREMENT THE PASS NUMBER
9792 056150 042767 100000 122462 BIC #100000,$PASS ;;DON'T ALLOW A NEG. NUMBER
9793 056156 005367 000122 DEC $EOPCT ;;LOOP?
9794 056162 003027 BGT $DOAGN ;;YES
9795 056164 016767 000116 000112 MOV $ENDCT,$EOPCT ;;RESTORE COUNTER
9796 056172 012767 056313 122350 MOV # $ENDMG,MSGADR
9797 056200 104401 TYPE ;;TYPE "END PASS #"
9798 056202 016746 122432 MOV $PASS,-(SP) ;;SAVE $PASS FOR TYPEOUT
9799 056206 104403 TYPOS ;;GO TYPE--OCTAL ASCII, NO LEADING ZEROES
9800 056210 006 .BYTE 6
9801 056211 000 .BYTE 0
9802 056212 016767 000072 122330 MOV $ENULL,MSGADR
9803 056220 104401 TYPE ;;TYPE A NULL CHARACTER
9804 056222 013700 000042 $GET42: MOV @#42,RO ;;GET MONITOR ADDRESS
9805 056226 001405 BEQ $DOAGN ;;BRANCH IF NO MONITOR
9806 056230 000005 RESET ;;CLEAR THE WORLD

```

END OF PASS ROUTINE

```

9807 056232 004710          $ENDAD: JSR      PC,(R0)          ;;GO TO MONITOR
9808 056234 000240          NOP                          ;;SAVE ROOM
9809 056236 000240          NOP                          ;;FOR
9810 056240 000240          NOP                          ;;ACT11
9811 056242                $DOAGN:
9812 056242 013737 000004 000476 MOV      @#4,@#TEMPO      ;;SAVE CONTENTS OF LOCATION 4
9813 056250 012737 056266 000004 MOV      #1,@#4          ;;SET UP VECTOR IN CASE OF TRAP
9814 056256 012737 000001 164000 MOV      #1,@#164000     ;;NOTIFY MULTI-TESTER OF PASS COMPLETE
9815 056264 000402                BR      2$              ;;NO TRAP --> DO NOT NEED TO RESTORE STACK
9816 056266 062706 000004 1$: ADD      #4,SP          ;;RESET STACK BECAUSE OF TRAP
9817 056272 013737 000476 000004 2$: MOV      @#TEMPO,@#4      ;;RESTORE CONTENTS OF LOCATION 4
9818 056300 000167 123334                JMP      BEGIN          ;;RETURN
9819 056304 000001          $EOPCT: .WORD      1
9820 056306 000001          $ENDCT: .WORD      1
9821 056310          377      377      000 $ENULL: .BYTE      -1,-1,0      ;;NULL CHARACTER STRING
9822 056313          015      012      105 $ENDMG: .ASCIZ    <15><12>/END PASS #/
          056316          116      104      040
          056321          120      101      123
          056324          123      040      043
          056327          000

9823
9824
9825          ;;*****
9826          ; TABLE OF SIGN/DIGIT COMBINATIONS FOR OVERPUNCH
9827          ;;*****
9828 056330          173          OVPNCH: .BYTE      173          ; +0
9829 056331          101          .BYTE      101          ; +1
9830 056332          102          .BYTE      102          ; +2
9831 056333          103          .BYTE      103          ; +3
9832 056334          104          .BYTE      104          ; +4
9833 056335          105          .BYTE      105          ; +5
9834 056336          106          .BYTE      106          ; +6
9835 056337          107          .BYTE      107          ; +7
9836 056340          110          .BYTE      110          ; +8
9837 056341          111          .BYTE      111          ; +9
9838 056342          175          .BYTE      175          ; -0
9839 056343          112          .BYTE      112          ; -1
9840 056344          113          .BYTE      113          ; -2
9841 056345          114          .BYTE      114          ; -3
9842 056346          115          .BYTE      115          ; -4
9843 056347          116          .BYTE      116          ; -5
9844 056350          117          .BYTE      117          ; -6
9845 056351          120          .BYTE      120          ; -7
9846 056352          121          .BYTE      121          ; -8
9847 056353          122          .BYTE      122          ; -9
9848
9849
9850          ;;*****
9851          .SBTTL UNEXPECTED TIME-OUT TRAP ROUTINE
9852          ;;*****
9853          ;*ANY UNEXPECTED TIME-OUT TRAP WILL END UP HERE
9854          ;*
9855
9856 056354 012767 062436 122166 TIMOUT: MOV      #TIMMSG,MSGADR      ;TYPE OUT ERROR MESSAGE
9857 056362 104401                TYPE
9858 056364 032767 000040 121460 BIT      #BITS,UFDMODE      ;ARE WE IN XXDP UFD          ;PA001
9859 056372 001403                BEQ      1$              ;NO, SO BRANCH              ;PA001

```

UNEXPECTED TIME-OUT TRAP ROUTINE

```

9860 056374 005000          CLR      RO          ;CLEAR JUNK ARGUMENT      ;PA001
9861 056376 004767 001450  JSR      PC,ABORT    ;SYSMAC ABORT ROUTINE   ;PA001
9862 056402 000000          1$:     HALT
9863 056404 000002          RTI

```

```

9864
9865
9866      ;*****
9867      .SBTTL  ILLEGAL/RESERVED INSTRUCTION TRAP ROUTINE
9868      ;*****
9869      ;*ANY ILLEGAL OR RESERVED INSTRUCTION TRAPS WILL END UP HERE.
9870      ;*

```

```

9871
9872 056406 012767 062504 122134 ILLRES: MOV      #ILLMSG,MSGADR ;TYPE OUT ERROR MESSAGE
9873 056414 104401          TYPE
9874 056416 032767 000040 121426 BIT      #BITS,UFDMODE ;ARE WE IN XXDP UFD      ;PA001
9875 056424 001403          BEQ      1$ ;NO, SO BRANCH           ;PA001
9876 056426 005000          CLR      RO          ;CLEAR JUNK ARGUMENT      ;PA001
9877 056430 004767 001416  JSR      PC,ABORT    ;SYSMAC ABORT ROUTINE   ;PA001
9878 056434 000000          1$:     HALT
9879 056436 000002          RTI

```

```

9880
9881
9882      ;*****
9883      .SBTTL  CLEAR REGISTERS R0 - R5 ROUTINE
9884      ;*****
9885      ;*THIS ROUTINE CLEARS GENERAL PURPOSE REGISTERS R0 THROUGH R5.
9886      ;*

```

```

9887
9888 ROUTINE CLRREG
9889 056440          CLRREG:
9890 056440 005000          LET R0 := #0          CLR      R0
9891 056442 005001          LET R1 := #0          CLR      R1
9892 056444 005002          LET R2 := #0          CLR      R2
9893 056446 005003          LET R3 := #0          CLR      R3
9894 056450 005004          LET R4 := #0          CLR      R4
9895 056452 005005          LET R5 := #0          CLR      R5
9896 056454          ENDRTN
9897 056454          $52442:
9898 056454          $52443:
9899 056454 000207          RTS      PC

```

```

9900      ;*****
9901      .SBTTL  CLEAR BUFFER ROUTINE
9902      ;*****
9903      ;*THIS ROUTINE CLEARS A 100 BYTE BUFFER WHOSE STARTING ADDRESS
9904      ;*HAS BEEN PUT INTO 'BFADRS'.

```

```

9905 ROUTINE CLRBUF
9906 056456          CLRBUF:
9907 056456          LET SAVRO := RO

```

CLEAR BUFFER ROUTINE

9906	056456	010067	122116		MOV	RO,SAVRO
	056462			LET SAVR1 := R1		
9907	056462	010167	122114		MOV	R1,SAVR1
	056466			LET RO := BFADRS		
9908	056466	016700	122050		MOV	BFADRS,RO
	056472			LET R1 := #40		
9909	056472	012701	000040		MOV	#40,R1
	056476			REPEAT		
9910	056476				\$52446:	
	056476	005020		LET (RO)+ := #0	CLR	(RO)+
9911	056500					
	056500	005301		LET R1 := R1 - #1	DEC	R1
9912	056502			UNTIL R1 EQ #0		
	056502	005701			TST	R1
	056504	001374			BNE	\$52446
9913	056506			LET RO := SAVRO		
	056506	016700	122066		MOV	SAVRO,RO
9914	056512			LET R1 := SAVR1		
	056512	016701	122064		MOV	SAVR1,R1
9915	056516			ENDRTN		
	056516				\$52444:	
	056516				\$52445:	
	056516	000207			RTS	PC
9916						
9917				;;*****		
9918				.SBTTL SET BUFFER ROUTINE		
9919				;;*****		
9920				;*THIS ROUTINE SETS ALL BITS IN A 100 BYTE BUFFER WHOSE STARTING		
9921				;*ADDRESS HAS BEEN PUT INTO 'BFADRS'.		
9922				;*		
9923						
9924	056520			ROUTINE SETBUF		
	056520				SETBUF:	
9925	056520			LET SAVRO := RO		
	056520	010067	122054		MOV	RO,SAVRO
9926	056524			LET SAVR1 := R1		
	056524	010167	122052		MOV	R1,SAVR1
9927	056530			LET RO := BFADRS		
	056530	016700	122006		MOV	BFADRS,RO
9928	056534			LET R1 := #40		
	056534	012701	000040		MOV	#40,R1
9929	056540			REPEAT		
	056540				\$52451:	
9930	056540			LET (RO)+ := #177777		
	056540	012720	177777		MOV	#177777,(RO)+
9931	056544			LET R1 := R1 - #1		
	056544	005301			DEC	R1
9932	056546			UNTIL R1 EQ #0		
	056546	005701			TST	R1
	056550	001373			BNE	\$52451
9933	056552			LET RO := SAVRO		
	056552	016700	122022		MOV	SAVRO,RO
9934	056556			LET R1 := SAVR1		
	056556	016701	122020		MOV	SAVR1,R1
9935	056562			ENDRTN		
	056562				\$52447:	

SET BUFFER ROUTINE

```

056562                                $52450:
056562 000207                          RTS    PC
9936
9937
9938 ;*****
9939 .SBTTL  ONES AND ZEROES BUFFER ROUTINE
9940 ;*****
9941 ;*THIS ROUTINE SETS AN ALTERNATING ONES AND ZEROES PATTERN IN A 100 BYTE
9942 ;*BUFFER WHOSE STARTING ADDRESS HAS BEEN PUT INTO 'BFADRS'.
9943 ;*
9944
9945 ROUTINE  ALTBUF
056564                                ALTBUF:
056564                                LET SAVRO := RO
9946 056564 010067 122010                MOV    RO,SAVRO
056564                                LET SAVR1 := R1
9947 056570 010167 122006                MOV    R1,SAVR1
056570                                LET RO := BFADRS
9948 056574 016700 121742                MOV    BFADRS,RO
056574                                LET R1 := #40
9949 056600 012701 000040                MOV    #40,R1
056600                                REPEAT
9950 056604                                $52454:
056604                                LET (RO)+ := #125252
9951 056604 012720 125252                MOV    #125252,(RO)+
056604                                LET R1 := R1 - #1
9952 056610 005301                DEC    R1
056610                                UNTIL R1 EQ #0
9953 056612 005701                TST   R1
056612 001373                BNE   $52454
9954 056616 016700 121756                MOV    SAVRO,RO
056616                                LET R1 := SAVR1
9955 056622 016701 121754                MOV    SAVR1,R1
056622                                ENDRTN
9956 056626                                $52452:
056626                                $52453:
056626 000207                          RTS    PC
9957
9958 ;*****
9959 .SBTTL  ONES AND ZEROES BUFFER ROUTINE FOR 2000 BYTE BUFFER
9960 ;*****
9961 ;*THIS ROUTINE SETS AN ALTERNATING ONES AND ZEROES PATTERN IN A 2000 BYTE
9962 ;*BUFFER WHOSE STARTING ADDRESS HAS BEEN PUT INTO 'BFADRS'.
9963 ;*
9964
9965 ROUTINE  ALTBUI
056630                                ALTBUI:
056630                                LET SAVRO := RO
9966 056630 010067 121744                MOV    RO,SAVRO
056630                                LET SAVR1 := R1
9967 056634 010167 121742                MOV    R1,SAVR1
056634                                LET RO := BFADRS
9968 056640 016700 121676                MOV    BFADRS,RO
056640                                LET R1 := #1000
9969 056644 012701 001000                MOV    #1000,R1
056644                                REPEAT
9970 056650

```

ONES AND ZEROS BUFFER ROUTINE FOR 2000 BYTE BUFFER

```

056650
9971 056650          LET (R0) := #125252
056650 012720 125252
9972 056654          LET R1 := R1 - #1
056654 005301
9973 056656          UNTIL R1 EQ #0
056656 005701
056660 001373
9974 056662          LET R0 := SAVRO
056662 016700 121712
9975 056666          LET R1 := SAVR1
056666 016701 121710
9976 056672          ENDRTN
056672
056672
056672 000207
9977

```

```

$52457:
MOV    #125252,(R0)
DEC    R1
TST    R1
BNE    $52457
MOV    SAVRO,R0
MOV    SAVR1,R1

$52455:
$52456:
RTS    PC

```

ONES AND ZEROES BUFFER ROUTINE FOR 2000 BYTE BUFFER

```

9979 ;:*****
9980 .SBTTL CLEAR DECIMAL STRING BUFFERS ROUTINE
9981 ;:*****
9982 ;*THIS ROUTINE CLEARS ALL BITS IN THE FOUR 40-BYTE DECIMAL
9983 ;*STRING BUFFERS.
9984 ;*
9985
9986 ROUTINE CLRDSB
                                CLRDSB:
056674 LET SAVRO := R0
                                MOV R0,SAVRO
9987 056674 010067 121700
                                LET SAVR1 := R1
                                MOV R1,SAVR1
056674 010167 121676
                                LET RO := #DSBUF1-1
                                MOV #DSBUF1-1,RO
9988 056700
                                LET R1 := #207
                                MOV #207,R1
056700 010167 121676
                                REPEAT
                                $52462:
9989 056704
                                LET (RO)+ :B= #0
                                CLRB (RO)+
056704 012700 064437
                                LET R1 := R1 #1
                                DEC R1
9990 056710
                                UNTIL R1 EQ #0
                                TST R1
056710 012701 000207
                                LET RO := SAVRO
                                BNE $52462
9991 056714
                                LET R1 := SAVR1
                                MOV SAVRO,RO
056714
                                ENDRTN
                                MOV SAVR1,R1
9992 056714 105020
                                $52460:
9993 056716 005301
                                $52461:
9994 056720
                                RTS PC
056720 005701
056722 001374
9995 056724
10000 ;:*****
10001 .SBTTL SAVE REGISTERS R0 - R5
10002 ;:*****
10003 ;*THIS ROUTINE SAVES R0 THROUGH R5 IN LOCATIONS SAVRO - SAVR5.
10004 ;*
10005 ROUTINE SAVREG
                                SAVREG:
056736 LET SAVRO := R0
                                MOV R0,SAVRO
10006 056736 010067 121636
                                LET SAVR1 := R1
                                MOV R1,SAVR1
056736 010167 121634
                                LET SAVR2 := R2
                                MOV R2,SAVR2
10007 056742
                                LET SAVR3 := R3
                                MOV R3,SAVR3
056742 010167 121634
                                LET SAVR4 := R4
                                MOV R4,SAVR4
10008 056746
                                LET SAVR5 := R5
                                MOV R5,SAVR5
056746 010267 121632
                                ENDRTN
056752
                                $52463:
10009 056752 010367 121630
10010 056756 010467 121626
10011 056762 010567 121624
10012 056766

```



D10

SAVE REGISTERS R0 - R5

056766  
056766 000207

10013

#52464: RTS PC

SAVE REGISTERS R0 - R5

```

10015 ;:*****
10016 .SBTTL RESTORE REGISTERS R0 R5
10017 ;:*****
10018 ;*THIS ROUTINE RESTORES R0 THROUGH R5 FROM LOCATIONS SAVR0 SAVR5
10019 ;*
10020
10021 056770 ROUTINE RESREG
10022 056770 RESREG:
056770 LET R0 := SAVR0
10023 056774 016700 121604 MOV SAVR0,R0
056774 LET R1 := SAVR1
10024 057000 016701 121602 MOV SAVR1,R1
057000 LET R2 := SAVR2
10025 057004 016702 121600 MOV SAVR2,R2
057004 LET R3 := SAVR3
10026 057010 016703 121576 MOV SAVR3,R3
057010 LET R4 := SAVR4
10027 057014 016704 121574 MOV SAVR4,R4
057014 LET R5 := SAVR5
10028 057020 016705 121572 MOV SAVR5,R5
057020 ENDRTN
057020 $52465:
057020 $52466:
057020 000207 RTS PC

10029
10030 ;:*****
10031 .SBTTL MEMORY MANAGEMENT SET UP ROUTINE
10032 ; ROUTINE WILL DETERMINE IF MEMORY MANAGEMENT SHOULD BE ENABLED OR DISABLED.
10033 ; IF MEMORY MANAGEMENT IS TO BE ENABLED THE APRS WILL BE MAPPED SUCH THAT THE
10034 ; VIRTUAL ADDRESS WILL BE MAPPED TO ITSELF AND MEMORY MANAGEMENT IS ENABLED.
10035 ;:*****
10036
10037 057022 ROUTINE MMSETUP
057022 MMSETUP:

10038
10039 ;+
10040 ; COMPLIMENT THE MEMORY MANAGEMENT CYCLE FLAG
10041 ; MMCYL = 0 -KTOFF
10042 ; MMCYL = -1 -KTON
10043 ;-
10044
10045 057022 LET @#MMCYL := COMP @#MMCYL
057022 005137 000446 COM @#MMCYL

10046
10047 ;+
10048 ; DETERMINE WHETHER TO ONLY CYCLE WITH MEMORY MANAGEMENT ON OR TO CYCLE
10049 ; WITH MEMORY MANAGEMENT OFF.
10050 ;-
10051
10052 057026 IF #BIT06 SETIN @SWR THEN
057026 032777 000100 121372 BIT #BIT06,@SWR
057034 001404 BEQ $52471

10053 057036 LET @#MMCYL := # 1
057036 012737 177777 000446 MOV #-1,@#MMCYL

10054 057044 ELSE
057044 000406 BR $52472
057046 $52471:

```

## MEMORY MANAGEMENT SET UP ROUTINE

```

10055 057046          IF #BIT07 SETIN @SWR THEN
      057046 032777 000200 121352          BIT    @BIT07,@SWR
      057054 001402          BEQ    $52473
10056 057056          LET @#MMCYL := #0
      057056 005037 000446          CLR    @#MMCYL
10057 057062          ENDIF
      057062          $52473:
10058 057062          ENDIF
      057062          $52472:
10059
10060
10061          ;+
10062          ; ONLY SET UP MEMORY MANAGEMENT REGISTERS IF 1ST PASS
10063          ;
10064 057062          IF @#PASS EQ #0 THEN
      057062 005737 000640          TST    @#PASS
      057066 001057          BNE    $52474
10065 057070          LET @#KIPAR0 := #0
      057070 005037 172340          CLR    @#KIPAR0
10066 057074          LET @#KIPAR1 := #200
      057074 012737 000200 172342          MOV    #200,@#KIPAR1
10067 057102          LET @#KIPAR2 := #400
      057102 012737 000400 172344          MOV    #400,@#KIPAR2
10068 057110          LET @#KIPAR3 := #600
      057110 012737 000600 172346          MOV    #600,@#KIPAR3
10069 057116          LET @#KIPAR4 := #1000
      057116 012737 001000 172350          MOV    #1000,@#KIPAR4
10070 057124          LET @#KIPAR5 := #1200
      057124 012737 001200 172352          MOV    #1200,@#KIPAR5
10071 057132          LET @#KIPAR6 := #1400
      057132 012737 001400 172354          MOV    #1400,@#KIPAR6
10072 057140          LET @#KIPAR7 := #7600
      057140 012737 007600 172356          MOV    #7600,@#KIPAR7
10073
10074 057146          LET @#KIPDR0 := #77406
      057146 012737 077406 172300          MOV    #77406,@#KIPDR0
10075 057154          LET @#KIPDR1 := #77406
      057154 012737 077406 172302          MOV    #77406,@#KIPDR1
10076 057162          LET @#KIPDR2 := #77406
      057162 012737 077406 172304          MOV    #77406,@#KIPDR2
10077 057170          LET @#KIPDR3 := #77406
      057170 012737 077406 172306          MOV    #77406,@#KIPDR3
10078 057176          LET @#KIPDR4 := #77406
      057176 012737 077406 172310          MOV    #77406,@#KIPDR4
10079 057204          LET @#KIPDR5 := #77406
      057204 012737 077406 172312          MOV    #77406,@#KIPDR5
10080 057212          LET @#KIPDR6 := #77406
      057212 012737 077406 172314          MOV    #77406,@#KIPDR6
10081 057220          LET @#KIPDR7 := #77406
      057220 012737 077406 172316          MOV    #77406,@#KIPDR7
10082 057226          ENDIF
      057226          $52474:
10083
10084          ;+
10085          ; NOW TURN ON MEMORY MANAGEMENT IF IN MEMORY MANAGEMENT MEMORY MANAGEMENT CYCLE
10086          ;
10087

```

## MEMORY MANAGEMENT SET UP ROUTINE

```

10088 057226          IF @#MMCYL NE #0 THEN
      057226 005737 000446          TST @#MMCYL
      057232 001404          BEQ $52475
10089 057234          LET @#SR1 := @#SR1 SET.BY #BIT0
      057234 052737 000001 177572  BIS @BIT0,@#SR1
10090
10091          ;*
10092          ; ELSE TURN OFF MEMORY MANAGEMENT
10093          ; -
      057242          ELSE
      057242 000403          BR $52475
      057244
10094 057244          LET @#SR1 := @#SR1 CLR.BY #BIT0
      057244 042737 000001 177572  BIC @BIT0,@#SR1
10095 057252          ENDIF
      057252          $52476:
10096 057252          ENDRTN
      057252          $52467:
      057252          $52470:
      057252 000207          RTS PC
10097

```

MEMORY MANAGEMENT SET UP ROUTINE

```

10099
10100 ;*****
10101 .SBTTL PACK - PACK DECIMAL STRING ROUTINE
10102 ;*****
10103
10104 ;*
10105 ; FUNCTIONAL DESCRIPTION
10106 ;
10107 ; THE PACK ROUTINE ACCEPTS UNPACKED DATA FROM A TEST CASE
10108 ; TABLE AND PACKS IT (TWO BCD CHARS/BYTE) INTO ONE OF THE
10109 ; DECIMAL STRING BUFFERS.
10110 ;
10111 ; INPUTS:
10112 ; 1. ADDRESS OF THE DESTINATION BUFFER. ** NOTE: ** THIS MUST
10113 ; BE #DSBUF1, #DSBUF2, OR #DS.EXP.
10114 ; 2. POINTER TO SOURCE DATA IN TEST CASE TABLE. ** NOTE: **
10115 ; THIS MUST BE DATA01, DATA03, OR DATA05.
10116 ; 3. POINTER TO DESTINATION STRING LENGTH. ** NOTE: ** THIS
10117 ; MUST BE DATA00, DATA02, OR DATA04.
10118 ;
10119 ; OUTPUTS:
10120 ; NONE
10121 ;
10122 ; PATHOLOGICAL CONNECTIONS:
10123 ; NONE.
10124 ;
10125 ; SUBORDINATE ROUTINES:
10126 ; NONE.
10127 ;
10128 ; FUNCTIONAL SIDE-EFFECTS.
10129 ; NONE.
10130 ;
10131 ; CALLING SEQUENCE:
10132 ;
10133 ; CALL PACK IN <DSTADR, SRCADR, LENGTH>. FOR INPUT PARAMETERS
10134 ; REFER TO ABOVE INPUTS:.
10135 ;
10136 ; NOTE:
10137 ; R3 IS USED AS A BYTE ASSEMBLY AREA.
10138 ; R4 IS USED TO HOLD THE SIGN OF THE STRING (+ = #14, = #15).
10139 ;
10140 ;
10141 057254 ROUTINE PACK <DSTADR, SRCADR, LENGTH>
10142 057254 PACK:
10143 057254 016500 000000 LET R0 := DSTADR(R5) MOV DSTADR(R5), R0
10144 057260 016501 000002 LET R1 := SRCADR(R5) MOV SRCADR(R5), R1
10145 057264 016502 000004 LET R2 := LENGTH(R5) CLR.BY #70000 MOV LENGTH(R5), R2
10146 057270 042702 070000 BIC #70000, R2
10147 ;*
10148 ; STORE SIGN OF STRING IN R4
10149 ;
10150 057274 IF (R1)+ EQ #0 THEN ;SIGN IS +

```

PACK - PACK DECIMAL STRING ROUTINE

```

057274 005721                                TST      (R1)+
057276 001003                                BNE      $52501
10151 057300                                LET R4 := #14
057300 012704 000014                            MOV      #14,R4
10152 057304                                ELSE ;SIGN IS -
057304 000402                                BR       $52502
057306                                $52501:
10153 057306                                LET R4 := #15
057306 012704 000015                            MOV      #15,R4
10154 057312                                ENDIF
057312                                $52502:
10155
10156 ;*
10157 ; IF LENGTH IS ZERO, THEN SIGN STRING AND EXIT; ELSE IF
10158 ; STRING LENGTH IS EVEN, STORE THE MSD (STRING LENGTH IS
10159 ; NOW ODD).NOW STORE ALL BUT THE LAST DIGIT. BUILD LSD
10160 ; & SIGN AND EXIT.
10161 ;-
10162
10163 057312                                IF R2 EQ #0 THEN ;STRING LENGTH = 0
057312 005702                                TST      R2
057314 001002                                BNE      $52503
10164 057316                                LET (R0)+ := R4 ;SIGN THE BUFFER AND EXIT
057316 110410                                MOV      R4,(R0)
10165 057320                                ELSE
057320 000425                                BR       $52504
057322                                $52503:
10166 057322                                IF #BIT00 NOTSET IN R2 THEN ;LENGTH IS EVEN & > 0
057322 032702 000001                            BIT      #BIT00,R2
057326 001002                                BNE      $52505
10167 057330                                LET (R0)+ := (R1)+ ;MOVE THE EVEN NIBBLE
057330 112120                                MOV      (R1)+,(R0)+
10168 057332                                LET R2 := R2 #1 ;DEC NIBBLE COUNTER
057332 005302                                DEC      R2
10169 057334                                ENDIF
057334                                $52505:
10170 057334                                WHILE R2 NE #1 DO
057334                                $52506:
057334 020227 000001                            CMP      R2,#1
057340 001410                                BEQ      $52507
10171 057342                                LET R3 := (R1)+ ;GET A BCD
057342 112103                                MOV      (R1)+,R3
10172 057344                                INLINE <ASH #4, R3> ;SHIFT IT TO THE HIGH NIBBLE
057344 072327 000004                            ASH      #4, R3
10173 057350                                LET R3 := R3 SET.BY (R1)+ ;PACK IN LOW NIBBLE
057350 152103                                BIS      (R1)+,R3
10174 057352                                LET (R0)+ := R3 ;STORE RESULT
057352 110320                                MOV      R3,(R0)+
10175 057354                                LET R2 := R2 - #2 ;DEC NIBBLE COUNTER
057354 162702 000002                            SUB      #2,R2
10176 057360                                ENDDO
057360 000765                                BR       $52506
057362                                $52507:
10177 057362                                LET R3 := (R1) ;GET LAST BCD
057362 111103                                MOV      (R1),R3
10178 057364                                INLINE <ASH #4, R3> ;SHIFT IT TO THE HIGH NIBBLE
057364 072327 000004                            ASH      #4, R3

```

PACK - PACK DECIMAL STRING ROUTINE

```
10179 057370          LET R3 :B= R3 SET.BY R4      ;ADD SIGN
      057370 150403
10180 057372          LET (R0) :B= R3        ;STORE LSD & SIGN
      057372 110310
10181 057374          ENDIF
      057374
10182 057374          ENDRTN
      057374
      057374
      057374 000207
10183
```

\$52504:  
\$52477:  
\$52500: RTS PC

PACK PACK DECIMAL STRING ROUTINE

```

10185 ;:*****
10186 .SBTTL SCOPE HANDLER ROUTINE
10187 ;:*****
10188 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
10189 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
10190 ;*AND LOAD THE ERROR FLAG (ERRFLG) INTO DISPLAY<15:08>
10191 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
10192 ;*SW14=1 LOOP ON TEST
10193 ;*SW11=1 INHIBIT ITERATIONS
10194 ;*
10195 ;*CALL
10196 ;* SCOPE ;:SCOPE=IOT
10197
10198 057376 $SCOPE:
10199 057376 104406 CKSWR ;:TEST FOR CHANGE IN SOFT SWR
10200 057400 032777 040000 121020 BIT @BIT14,@SWR ;:LOOP ON PRESENT TEST?
10201 057406 001034 BNF $OVER ;:YES IF SW14=1
10202 057410 032777 004000 121010 BI @BIT11,@SWR ;:INHIBIT ITERATIONS?
10203 057416 001011 BNE 1$ ;:BRANCH IF YES
10204 057420 005767 121214 TST $PASS ;:IF FIRST PASS OF PROGRAM
10205 057424 001406 BEQ 1$ ;: INHIBIT ITERATIONS
10206 057426 005267 120754 INC $ICNT ;:INCREMENT ITERATION COUNT
10207 057432 026767 120746 120746 CMP $TIMES,$ICNT ;:CHECK THE NUMBER OF ITERATIONS MADE
10208 057440 002017 BGE $OVER ;:BRANCH IF MORE ITERATION REQUIRED
10209 057442 012767 000001 120736 1$: MOV @1,$ICNT ;:REINITIALIZE THE ITERATION COUNTER
10210 057450 016767 000040 120726 MOV $MXCNT,$TIMES ;:SET NUMBER OF ITERATIONS TO DO
10211 057456 005267 120720 $SVLAD: INC $TSTNM ;:COUNT TEST NUMBERS
10212 057462 016767 120714 121146 MOV $TSTNM,$TESTN ;:SET TEST NUMBER IN APT MAILBOX
10213 057470 011667 120714 MOV (SP),$LPADR ;:SAVE SCOPE LOOP ADDRESS
10214 057474 005067 120756 CLR ERRFLG ;:ZERO THE ERROR FLAG
10215 057500 016777 120676 120722 $OVER: MOV $TSTNM,@DISPLAY ;:DISPLAY TEST NUMBER
10216 057506 016716 120676 MOV $LPADR,(SP) ;:FUDGE RETURN ADDRESS
10217 057512 000002 RTI ;:FIXES PS
10218 057514 000100 $MXCNT: .WORD 100 ;:MAXIMUM NUMBER OF ITERATIONS
10219
10220
10221 ;:*****
10222 .SBTTL POWER DOWN AND UP ROUTINES
10223 ;:*****
10224 ;POWER DOWN ROUTINE
10225 057516 012737 057666 000024 $PWRDN: MOV @ILLUP,@PWRVEC ;:SET FOR FAST UP
10226 057524 012737 000340 000026 MOV @340,@PWRVEC+2 ;:PRIO:7
10227 057532 010046 MOV R0,-(SP) ;:PUSH R0 ON STACK
10228 057534 010146 MOV R1,-(SP) ;:PUSH R1 ON STACK
10229 057536 010246 MOV R2,-(SP) ;:PUSH R2 ON STACK
10230 057540 010346 MOV R3,-(SP) ;:PUSH R3 ON STACK
10231 057542 010446 MOV R4,-(SP) ;:PUSH R4 ON STACK
10232 057544 010546 MOV R5,-(SP) ;:PUSH R5 ON STACK
10233 057546 017746 120654 MOV @SWR,-(SP) ;:PUSH @SWR ON STACK
10234 057552 010667 000114 MOV SP,$SAVR6 ;:SAVE SP
10235 057556 012737 057570 000024 MOV @PWRUP,@PWRVEC ;:SET UP VECTOR
10236 057564 000000 HALT
10237 057566 000776 BR -2 ;:HANG UP
10238
10239 ;:*****
10240 ;POWER UP ROUTINE
10241 057570 012737 057666 000024 $PWRUP: MOV @ILLUP,@PWRVEC ;:SET FOR FAST DOWN

```



POWER DOWN AND UP ROUTINES

```

10242 057576 016706 000070      MOV      $SAVR6,SP      ;;GET SP
10243 057602 005067 000064      CLR      $SAVR6        ;;WAIT LOOP FOR THE TTY
10244 057606 005267 000060      1$: INC      $SAVR6        ;;WAIT FOR THE INC
10245 057612 001375                BNE      1$            ;;OF WORD
10246 057614 012677 120606      MOV      (SP)+,@SWR    ;;POP STACK INTO @SWR
10247 057620 012605                MOV      (SP)+,R5      ;;POP STACK INTO R5
10248 057622 012604                MOV      (SP)+,R4      ;;POP STACK INTO R4
10249 057624 012603                MOV      (SP)+,R3      ;;POP STACK INTO R3
10250 057626 012602                MOV      (SP)+,R2      ;;POP STACK INTO R2
10251 057630 012601                MOV      (SP)+,R1      ;;POP STACK INTO R1
10252 057632 012600                MOV      (SP)+,R0      ;;POP STACK INTO R0
10253 057634 012737 057516 000024  MOV      #$PWRDN,@PWRVEC ;;SET UP THE POWER DOWN VECTOR
10254 057642 012737 000340 000026  MOV      #340,@PWRVEC+2 ;;PRIO:7
10255 057650 012767 062600 120672  MOV      #PWRMSG,MSGADR
10256 057656 104401                TYPE
10257 057660 012716                MOV      (PC)+,(SP)    ;;RESTART AT PHALT
10258 057662 057674      $PWRAD: .WORD  PHALT    ;;RESTART ADDRESS
10259 057664 000002                RTI
10260 057666 000000      $ILLUP: HALT          ;;THE POWER UP SEQUENCE WAS STARTED
10261 057670 000776                BR      -2            ;; BEFORE THE POWER DOWN WAS COMPLETE
10262 057672 000000      $SAVR6: 0            ;;PUT THE SP HERE
10263
10264 057674
10265 057674 000137 001640      PHALT:  JMP      @#BEGIN          :RESTART THE TEST AT THE BEGINNING
10266
10267      ;;*****
10268      .SBTTL  ERROR HANDLER ROUTINE
10269      ;;*****
10270      ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT.
10271      ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
10272      ;*AND GO TO $ERRTYP ON ERROR
10273      ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
10274      ;*SW15=1      HALT ON ERROR
10275      ;*SW13=1      INHIBIT ERROR TYPEOUTS
10276
10277 057700      ERROR:
10278 057700 104406                CKSWR
10279 057702 005267 120550      7$: INC      ERRFLG      ;;TEST FOR CHANGE IN SOFT SWR
10280 057706 001775                BEQ      7$            ;;SET THE ERROR FLAG
10281 057710 016777 120466 120512  MOV      $TSTNM,@DISPLAY ;;DON'T LET THE FLAG GO TO ZERO
10282 057716 005267 120472      1$: INC      $ERTTL      ;;DISPLAY TEST NUMBER AND ERROR FLAG
10283 057722 011667 120472      MOV      (SP), $ERRPC   ;;COUNT THE NUMBER OF ERRORS
10284 057726 162767 000004 120464  SUB      #4, $ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
10285 057734 032777 020000 120464  BIT      #BIT13,@SWR    -:SKIP TYPEOUT IF SET
10286 057742 001006                BNE      20$          ;;SKIP TYPEOUTS
10287 057744 004767 000242      JSR      PC, $ERRTYP    ;;GO TO USER ERROR ROUTINE
10288 057750 012767 000443 120572  MOV      #$CRLF,MSGADR
10289 057756 104401                TYPE
10290 057760      20$:
10291 057760 122767 000001 120664  CMPB     #APTENV,$ENV   ;;RUNNING IN APT MODE
10292 057766 001007                BNE      21$          ;;NO,SKIP APT ERROR REPORT
10293 057770 116767 120556 000004  MOVB     ERR,21$       ;;SET ITEM NUMBER AS ERROR NUMBER
10294 057776 004767 002110      JSR      PC,$ATY4      ;;REPORT FATAL ERROR TO APT
10295 060002 000                21$: .BYTE 0
10296 060003 000                .BYTE 0
10297 060004 000777      22$: BR      22$        ;;APT ERROR LOOP
10298 060006 005777 120414      2$: TST      @SWR      ;;HALT ON ERROR

```

ERROR HANDLER ROUTINE

```

10299 060012 100011          BPL      3$          ;;SKIP IF CONTINUE
10300 060014 032767 000040 120030 BIT      @BIT5,UFDMODE ;;ARE WE IN UFD MONITOR? ;PA001
10301 060022 001403          BEQ      8$          ;;NO, SO BRANCH ;PA001
10302 060024 005000          CLR      R0         ;;CLEAR A POSSIBLE JUNK ARGUMENT ;PA001
10303 060026 004767 000020 JSR      PC,ABORT    ;;CALL THE XXDP ABORT ROUTINE ;PA001
10304 060032 000000          8$: HALT          ;;HALT ON ERROR!
10305 060034 104406          CKSWR          ;;TEST FOR CHANGE IN SOFT-SWR
10306 060036
10307 060036 022737 056232 000042 3$: CMP      @ENDAD,@#42 ;;ACT-11 AUTO-ACCEPT?
10308 060044 001001          BNE      6$          ;;BRANCH IF NO
10309 060046 000000          HALT          ;;YES
10310 060050
10311 060050 000207          6$: RTS      PC         ;;RETURN
10312
10313
10314
10315
10316
10317
10318
10319
10320
10321
10322
10323
10324 060052 005767 121214          .LIST  MEB ;PA001
060052 005767 121214          $ABORT
060056 001454          ABORT: TST      UFDPLG ;;TEST FOR USER FRIENDLY MODE
060060 020027 000032          BEQ      NOABRT    ;;IF NOT UFD THEN CONTINUE NORMAL OPERATION
060064 001443          CMP      R0,#32    ;;IS IT A +Z ?
060066 020027 000003          BEQ      ABORTZ    ;;JUST GO BACK TO CHAIN IF IT IS (NO ERROR)
060072 001404          CMP      R0,#3    ;;IS IS A +C ?
060074 005767 121174          BEQ      ABORTC    ;;BR TO LOAD +C ON XXDP+ STACK (NO ERROR)
060100 001443          TST      UQUIET    ;;TEST FOR USER-QUIET MODE
060102 000422          BEQ      NOABRT    ;;IF FIELD-SERVICE MODE, CONTINUE NORMAL OPERATION
060104 016767 121156 117716 BR       ABORTE     ;;SET DRSEERR THEN LEAVE
060112 016767 121152 117712 ABORTC: MOV      SAV30,30 ;;RESTORE EMT LOCATION (30)
060120 104043          MOV      SAV32,32 ;;RESTORE EMT PRIORITY LOCATION (32)
060122 005720          EMT      +43      ;;GET XXDP STACK LOC. INTO R0 FROM MONITOR
060124 001376          1$: TST      (R0)+ ;;FIND END OF STACK
060126 112760 000057 177777 BNE      1$
060134 112720 000136          MOVB     #' /,-1(R0) ;;LOAD SLASH OVER ZERO
060140 112720 000103          MOVB     #' +,(R0)+ ;;LOAD U^ARROW
060144 105010          MOVB     #' C,(R0)+ ;;LOAD C
060146 000412          CLRB    (R0)      ;;MAKE NEW END TO STACK
060150 016767 121112 117652 BR       ABORTZ    ;;NOW LEAVE
060156 016767 121106 117646 ABORTE: MOV      SAV30,30 ;;RESTORE EMT LOCATION (30)
060164 104042          MOV      SAV32,32 ;;RESTORE EMT PRIORITY LOCATION (32)
060166 012760 177777 000042 EMT      +42      ;;GET DCA LOCATION INTO R0 FROM MONITOR
060174 013700 000042          MOV      @-1,42(R0) ;;SET A -1 INTO LOCATION DRSEERR IN MONITOR
060200 005037 000042          ABORTZ: MOV      @#42,R0 ;;AND PUT THE MONITOR RETURN ADDRESS IN R0
060204 000167 176022          CLR      @#42     ;;CLEAR MONITOR RETURN FLAG
060210 000207          JMP      $ENDAD    ;;RETURN TO MONITOR-DO NOT PUSH STACK HERE
10325
10326
10327
10328          NOABRT: RTS      PC         ;;IF NOTUFD RETURN TO MAINLINE
          .NLIST MEB
;*****

```

ERROR MESSAGE TYPEOUT ROUTINE

```

10329 .SBTTL ERROR MESSAGE TYPEOUT ROUTINE
10330 ;*****
10331 ;*THIS ROUTINE USES THE "ERROR TYPE NUMBER" (ERR) TO DETERMINE WHICH
10332 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
10333 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
10334
10335 060212 $ERRTYP:
10336 060212 012767 062750 120330      MOV      #ERRHDR,MSGADR      ;TYPE ERROR HEADER
10337 060220 104401                      TYPE
10338 060222 016767 120324 120244      MOV      ERR,ERRTMP        ;TYPE APPROPRIATE ERROR MESSAGE
10339 060230 005367 120240              DEC      ERRTMP
10340 060234 006367 120234              ASL      ERRTMP
10341 060240 062767 060372 120226      ADD      #ERRTBL,ERRTMP
10342 060246 017767 120222 120274      MOV      @ERRTMP,MSGADR
10343 060254 104401                      TYPE
10344 060256 012767 062630 120264      MOV      #SPACE1,MSGADR    ;TYPE ONE SPACE
10345 060264 104401                      TYPE
10346 060266 016746 120126              MOV      $ERRPC,-(SP)      ;TYPE THE ERROR PC
10347 060272 104402                      TYPOC
10348 060274 016767 120246 120172      MOV      CTL,ERRTMP        ;TYPE FAILING CONTROL CHIP NUMBER
10349 060302 162767 000004 120164      SUB      #4,ERRTMP
10350 060310 006367 120160              ASL      ERRTMP
10351 060314 062767 060400 120152      ADD      #CTLTBL,ERRTMP
10352 060322 017767 120146 120220      MOV      @ERRTMP,MSGADR
10353 060330 104401                      TYPE
10354 060332 012767 000443 120210      MOV      #CRLF,MSGADR     ;TYPE CARRIAGE-RETURN/LINE-FEED
10355 060340 104401                      TYPE
10356 060342 005737 000446              TST      @MMCYL           ;IF MEMORY MANAGEMENT NOT ENABLED
10357 060346 001004                      BNE      1$              ; THEN TYPE OUT
10358 060350 012767 063136 120172      MOV      #MMDIS,MSGADR    ; MEMORY MANAGEMENT DISABLED MESSAGE
10359 060356 000403                      BR       2$
10360 060360 012767 063176 120162 1$:  MOV      #MENA,MSGADR     ; ELSE TYPE OUT MEMORY MANAGEMENT ENABLED
10361 060366 104401                      2$:  TYPE
10362 060370 000207                      RTS      PC
10363
10364 060372 063011      ERRTBL: .WORD  ERR1      ;ADDRESSES OF ERROR MESSAGES
10365 060374 063042      .WORD  ERR2
10366 060376 063071      .WORD  ERR3
10367
10368 060400 062632      CTLTBL: .WORD  CTL4      ;ADDRESSES OF CHIP NUMBER MESSAGES
10369 060402 062647      .WORD  CTL5
10370 060404 062664      .WORD  CTL6
10371 060406 062701      .WORD  CTL7
10372 060410 062716      .WORD  CTL8
10373 060412 062733      .WORD  CTL9
10374
10375
10376 ;*****
10377 .SBTTL TYPE ROUTINE
10378 ;*****
10379 ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
10380 ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
10381 ;*NOTE1:      $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
10382 ;*NOTE2:      $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
10383 ;*NOTE3:      $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
10384 ;*
10385

```

TYPE ROUTINE

```

10386 060414 032767 000100 117430 $TYPE: BIT #BIT6,UFDMODE ;;ARE WE IN QUIET XXDP? ;PA001
10387 060422 001032 BNE 3$ ;;YES SO EXIT ;PA001
10388 060424 105767 120005 TSTB $TPFLG ;;IS THERE A TERMINAL?
10389 060430 100002 BPL 1$ ;;BR IF YES
10390 060432 000000 HALT ;;HALT HERE IF NO TERMINAL
10391 060434 000425 BR 3$ ;;LEAVE
10392 060436 010046 1$: MOV RO,-(SP) ;;SAVE RO
10393 060440 016700 120104 MOV MSGADR,RO ;;GET ADDRESS OF ASCIZ STRING
10394 060444 122767 000001 120200 CMPB #APTENV,$ENV ;;RUNNING IN APT MODE
10395 060452 001006 BNE 62$ ;;NO,GO CHECK FOR APT CONSOLE
10396 060454 132767 000100 120171 BITB #APTPOOL,$ENVM ;;SPOOL MESSAGE TO APT
10397 060462 001402 BEQ 62$ ;;NO,GO CHECK FOR CONSOLE
10398 060464 004767 001412 JSF PC,$ATY3 ;;SPOOL MESSAGE TO APT
10399 060470 132767 000040 120155 62$: BITB #APTCSUP,$ENVM ;;APT CONSOLE SUPPRESSED
10400 060476 001003 JNE 60$ ;;YES,SKIP TYPE OUT
10401 060500 112046 2$: MOVB (RO)+,-(SP) ;;PUSH CHARACTER TO BE TYPED ONTO STACK
10402 060502 001003 BNE 4$ ;;BR IF IT ISN'T THE TERMINATOR
10403 060504 005726 TST (SP)+ ;;IF TERMINATOR POP IT OFF THE STACK
10404 060506 012600 60$: MOV (SP)+,RO ;;RESTORE RO
10405 060510 000002 3$: RTI ;;RETURN
10406 060512 122716 000011 4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
10407 060516 001432 BEQ 8$
10408 060520 122716 000200 CMPB #CRLF,(SP) ;;BRANCH IF NOT <CRLF>
10409 060524 001010 BNE 5$
10410 060526 005726 TST (SP)+ ;;POP <CR><LF> EQUIV
10411 060530 012767 000443 120012 MOV #CRLF,MSGADR
10412 060536 104401 TYPE ;;TYPE A CR AND LF
10413 060540 105067 000130 CLRB $CHARCNT ;;CLEAR CHARACTER COUNT
10414 060544 000755 BR 2$ ;;GET NEXT CHARACTER
10415 060546 004767 000056 5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
10416 060552 126726 117656 6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
10417 060556 001350 BNE 2$ ;;IF NO GO GET NEXT CHAR.
10418 060560 016746 117646 MOV $NULL,(SP) ;;GET # OF FILLER CHARS. NEEDED
10419 ;;AND THE NULL CHAR.
10420 060564 105366 000001 7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
10421 060570 002770 BLT 6$ ;;BR IF NO- GO POP THE NULL OFF OF STACK
10422 060572 004767 000032 JSR PC,$TYPEC ;;GO TYPE A NULL
10423 060576 105367 000072 DECB $CHARCNT ;;DO NOT COUNT AS A COUNT
10424 060602 000770 BR 7$ ;;LOOP
10425
10426 ;HORIZONTAL TAB PROCESSOR
10427
10428 060604 112716 000040 8$: MOVB #' ,(SP) ;;REPLACE TAB WITH SPACE
10429 060610 004767 000014 9$: JSR PC,$TYPEC ;;TYPE A SPACE
10430 060614 132767 000007 000052 BITB #7,$CHARCNT ;;BRANCH IF NOT AT
10431 060622 001372 BNE 9$ ;;TAB STOP
10432 060624 005726 TST (SP)+ ;;POP SPACE OFF STACK
10433 060626 000724 BR 2$ ;;GET NEXT CHARACTER
10434 060630 105737 177564 $TYPEC: TSTB @#TPS ;;WAIT UNTIL PRINTER IS READY
10435 060634 100375 BPL $TYPEC
10436 060636 116637 000002 177566 MOVB 2(SP),@#TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
10437 060644 122766 000015 000002 CMPB #CR,2(SP) ;;IS CHARACTER A CARRIAGE RETURN?
10438 060652 001003 BNE 1$ ;;BRANCH IF NO
10439 060654 105067 000014 CLRB $CHARCNT ;;YES--CLEAR CHARACTER COUNT
10440 060660 000406 BR $TYPEX ;;EXIT
10441 060662 122766 000012 000002 1$: CMPB #LF,2(SP) ;;IS CHARACTER A LINE FEED?
10442 060670 001402 BEQ $TYPEX ;;BRANCH IF YES

```

TYPE ROUTINE

```

10443 060672 105227          INCB      (PC)+          ;;COUNT THE CHARACTER
10444 060674 000000          $CHARCNT: .WORD 0          ;;CHARACTER COUNT STORAGE
10445 060676 000207          $TYPEX: RTS      PC
10446
10447          ;;*****
10448          .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
10449          ;;*****
10450          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
10451          ;*OCTAL (ASCII) NUMBER AND TYPE IT.
10452          ;*$TYPOS- -ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
10453          ;*CALL:
10454          ;*      MOV      NUM,-(SP)          ;;NUMBER TO BE TYPED
10455          ;*      TYPOS          ;;CALL FOR TYPEOUT
10456          ;*      .BYTE  N          ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
10457          ;*      .BYTE  M          ;;M=1 OR 0
10458          ;*                                  ;;1=TYPE LEADING ZEROS
10459          ;*                                  ;;0=SUPPRESS LEADING ZERGS
10460          ;*
10461          ;*$TYPON ---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
10462          ;*$TYPOS OR $TYPOC
10463          ;*CALL:
10464          ;*      MOV      NUM, (SP)          ;;NUMBER TO BE TYPED
10465          ;*      TYPON          ;;CALL FOR TYPEOUT
10466          ;*
10467          ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
10468          ;*CALL:
10469          ;*      MOV      NUM,-(SP)          ;;NUMBER TO BE TYPED
10470          ;*      TYPOC          ;;CALL FOR TYPEOUT
10471
10472 060700 017646 000000          $TYPOS: MOV      @ (SP), -(SP)          ;;PICKUP THE MODE
10473 060704 116667 000001 000227  MOVB     1(SP), $OFILL          ;;LOAD ZERO FILL SWITCH
10474 060712 112667 000225          MOVB     (SP)+, $OMODE+1          ;;NUMBER OF DIGITS TO TYPE
10475 060716 062716 000002          ADD      #2, (SP)          ;;ADJUST RETURN ADDRESS
10476 060722 000406          BR      $TYPON
10477
10478 060724 112767 000001 000207  $TYPOC: MOVB     #1, $OFILL          ;;SET THE ZERO FILL SWITCH
10479 060732 112767 000006 000203  MOVB     #6, $OMODE+1          ;;SET FOR SIX(6) DIGITS
10480
10481 060740 032767 000100 117104  $TYPON: BIT      #BIT6, UFDMODE          ;;ARE WE IN QUIET XXDP?          ;PA001
10482 060746 001401          BEQ      9$          ;;NO, SO PRINT          ;PA001
10483 060750 000002          RTI          ;;YES, SO JUST RETURN          ;PA001
10484 060752 112767 000005 000160  9$:  MOVB     #5, $OCNT          ;;SET THE ITERATION COUNT
10485 060760 010346          MOV      R3, -(SP)          ;;SAVE R3
10486 060762 010446          MOV      R4, -(SP)          ;;SAVE R4
10487 060764 010546          MOV      R5, -(SP)          ;;SAVE R5
10488 060766 116704 000151          MOVB     $OMODE+1, R4          ;;GET THE NUMBER OF DIGITS TO TYPE
10489 060772 005404          NEG      R4
10490 060774 062704 000006          ADD      #6, R4          ;;SUBTRACT IT FOR MAX. ALLOWED
10491 061000 110467 000136          MOVB     R4, $OMODE          ;;SAVE IT FOR USE
10492 061004 116704 000131          MOVB     $OFILL, R4          ;;GET THE ZERO FILL SWITCH
10493 061010 016605 000012          MOV      12(SP), R5          ;;PICKUP THE INPUT NUMBER
10494 061014 005003          CLR      R3          ;;CLEAR THE OUTPUT WORD
10495 061016 006105          1$:  ROL      R5          ;;ROTATE MSB INTO "C"
10496 061020 000404          BR      3$          ;;GO DO MSB
10497 061022 006105          2$:  ROL      R5          ;;FORM THIS DIGIT
10498 061024 006105          ROL      R5
10499 061026 006105          ROL      R5

```

BINARY TO OCTAL (ASCII) AND TYPE

```

10500 061030 010503          MOV      R5,R3
10501 061032 006103          ROL      R3          ;;GET LSB OF THIS DIGIT
10502 061034 105367 000102    3$:     DECB     $OMODE  ;;TYPE THIS DIGIT?
10503 061040 100020          BPL      7$          ;;BR IF NO
10504 061042 042703 177770    BIC      #177770,R3  ;;GET RID OF JUNK
10505 061046 001002          BNE      4$          ;;TEST FOR 0
10506 061050 005704          TST      R4          ;;SUPPRESS THIS 0?
10507 061052 001403          BEQ      5$          ;;BR IF YES
10508 061054 005204          4$:     INC      R4          ;;DON'T SUPPRESS ANYMORE 0'S
10509 061056 052703 000060    BIS      #'0,R3     ;;MAKE THIS DIGIT ASCII
10510 061062 052703 000040    5$:     BIS      #' ,R3  ;;MAKE ASCII IF NOT ALREADY
10511 061066 110367 000044    MOV      R3,8$     ;;SAVE FOR TYPING
10512 061072 012767 061136 117450  MOV      #8$,MSGADR
10513 061100 104401          TYPE     ;;GO TYPE THIS DIGIT
10514 061102 105367 000032    7$:     DECB     $OCNT  ;;COUNT BY 1
10515 061106 003345          BGT      2$          ;;BR IF MORE TO DO
10516 061110 002402          BLT      6$          ;;BR IF DONE
10517 061112 005204          INC      R4          ;;INSURE LAST DIGIT ISN'T A BLANK
10518 061114 000742          BR       2$          ;;GO DO THE LAST DIGIT
10519 061116 012605          6$:     MOV      (SP)+,R5  ;;RESTORE R5
10520 061120 012604          MOV      (SP)+,R4  ;;RESTORE R4
10521 061122 012603          MOV      (SP)+,R3  ;;RESTORE R3
10522 061124 016666 000002 000004  MOV      2(SP),4(SP) ;;SET THE STACK FOR RETURNING
10523 061132 012616          MOV      (SP)+,(SP)
10524 061134 000002          RTI     ;;RETURN
10525 061136 000          8$:     .BYTE   0          ;;STORAGE FOR ASCII DIGIT
10526 061137 000          .BYTE   0          ;;TERMINATOR FOR TYPE ROUTINE
10527 061140 000          $OCNT:  .BYTE   0          ;;OCTAL DIGIT COUNTER
10528 061141 000          $OFILL: .BYTE   0          ;;ZERO FILL SWITCH
10529 061142 000000          $OMODE: .WORD   0          ;;NUMBER OF DIGITS TO TYPE
10530
10531
10532          ;;*****
10533          .SBTTL  TTY INPUT ROUTINE
10534          ;;*****
10535          .ENABL  LSB
10536
10537          ;;*****
10538          ;*SOFTWARE SWITCH REGISTER CHANGE ROUTINE.
10539          ;*ROUTINE IS ENTERED FROM THE TRAP HANDLER, AND WILL
10540          ;*SERVICE THE TEST FOR CHANGE IN SOFTWARE SWITCH REGISTER TRAP CALL
10541          ;*WHEN OPERATING IN TTY FLAG MODE.
10542
10543 061144 022767 000176 117254  $CKSWR: CMP      #SWREG,SWR  ;;IS THE SOFT SWR SELECTED?
10544 061152 001134          BNE      15$        ;;BRANCH IF NO
10545 061154 105737 177560          TST      @#TKS     ;;CHAR THERE?
10546 061160 100131          BPL      15$        ;;IF NO, DON'T WAIT AROUND
10547 061162 113746 177562          MOV      @#TKB,(SP) ;;SAVE THE CHAR
10548 061166 042716 177600          BIC      #+C177,(SP) ;;STRIP-OFF THE ASCII
10549 061172 032767 000040 116652  BIT      #BITS,UFDMODE ;;ARE WE UNDER UFD?
10550 061200 001422          BEQ      21$        ;;NO, SO BRANCH
10551 061202 111600          MOV      (SP),R0   ;;YES, GET THE CHARACTER JUST TYPED
10552 061204 122700 000003          CMP      #3,R0     ;;IS IT A CONTROL C?
10553 061210 001005          BNE      22$        ;;NO, SO BRANCH
10554 061212 012767 062040 117330  MOV      # $CNTLC,MSGADR ;;YES, GET CONTROL C TO ECHO
10555 061220 104401          TYPE
10556 061222 000407          BR       23$        ;;NOW DO THE ABORT

```

TTY INPUT ROUTINE

```

10557 061224 122700 000132      22$:  CMPB   #132,RO      ;;IS IT A CONTROL Z?      ;PA001
10558 061230 001006                BNE   21$          ;;NO, SO BRANCH          ;PA001
10559 061232 012767 062045 117310  MOV   #$CNTLZ,MSGADR ;;YES, GET CONTROL Z TO ECHO ;PA001
10560 061240 104401                TYPE
10561 061242 004767 176604      23$:  JSR   PC,ABORT    ;;SEE IF WE NEED TO ABORT ;PA001
10562 061246 022726 000007      21$:  CMP   #7,(SP)+    ;;IS IT A CONTROL G?
10563 061252 001074                BNE   15$          ;;NO, RETURN TO USER
10564 061254 126727 117142 000001  CMPB  $AUTOB,#1    ;;ARE WE RUNNING IN AUTO MODE?
10565 061262 001470                BEQ   15$          ;;BRANCH IF YES
10566
10567 061264 012767 062033 117256  MOV   #$CNTLG,MSGADR
10568 061272 104401                TYPE      ;;ECHO THE CONTROL G (+G)
10569 061274 012767 062052 117246  $GTSWR: MOV  #$MSWR,MSGADR
10570 061302 104401                TYPE      ;;TYPE CURRENT CONTENTS
10571 061304 016746 116666  MOV   SWREG,-(SP)  ;;SAVE SWREG FOR TYPEOUT
10572 061310 104402                TYPOC    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
10573 061312 012767 062063 117230  MOV   #$MNEW,MSGADR
10574 061320 104401                TYPE      ;;PROMPT FOR NEW SWR
10575 061322 005046      19$:  CLR   (SP)        ;;CLEAR COUNTER
10576 061324 005046                CLR   -(SP)       ;;THE NEW SWR
10577 061326 105737 177560      7$:  TSTB  @#TKS       ;;CHAR THERE?
10578 061332 100375                BPL   7$          ;;IF NOT TRY AGAIN
10579
10580 061334 113746 177562  MOVB  @#TKB,-(SP)  ;;PICK UP CHAR
10581 061340 042716 177600                BIC   #7C177,(SP) ;;MAKE IT 7-BIT ASCII
10582
10583 061344 021627 000025      9$:  CMP   (SP),#25    ;;IS IT A CONTROL-U?
10584 061350 001007                BNE   10$         ;;BRANCH IF NOT
10585 061352 012767 062026 117170  MOV   #$CNTLU,MSGADR
10586 061360 104401                TYPE      ;;YES, ECHO CONTROL U (+U)
10587 061362 062706 000006      20$:  ADD   #6,SP       ;;IGNORE PREVIOUS INPUT
10588 061366 000755                BR    19$        ;;LET'S TRY IT AGAIN
10589
10590 061370 021627 000015      10$:  CMP   (SP),#15    ;;IS IT A <CR>?
10591 061374 001024                BNE   16$         ;;BRANCH IF NO
10592 061376 005766 000004                TST   4(SP)       ;;YES, IS IT THE FIRST CHAR?
10593 061402 001403                BEQ   11$         ;;BRANCH IF YES
10594 061404 016677 000002 117014  MOV   2(SP),@SWR  ;;SAVE NEW SWR
10595 061412 062706 000006      11$:  ADD   #6,SP       ;;CLEAR UP ^TACK
10596 061416 012767 000443 117124  14$:  MOV   #$CRLF,MSGADR
10597 061424 104401                TYPE      ;;ECHO <CR> AND <LF>
10598 061426 126727 116771 000001  CMPB  $INTAG,#1   ;;RE-ENABLE TTY KBD INTERRUPTS?
10599 061434 001003                BNE   15$         ;;BRANCH IF NOT
10600 061436 012737 000100 177560  MOV   #100,@#TKS ;;RE-ENABLE TTY KBD INTERRUPTS
10601 061444 000002      15$:  RTI                    ;;RETURN
10602 061446 004767 177156      16$:  JSR   PC,$TYPEC   ;;ECHO CHAR
10603 061452 021627 000060                CMP   (SP),#60    ;;CHAR < 0?
10604 061456 002420                BLT   18$         ;;BRANCH IF YES
10605 061460 021627 000067                CMP   (SP),#67    ;;CHAR > 7?
10606 061464 003015                BGT   18$         ;;BRANCH IF YES
10607 061466 042726 000060                BIC   #60,(SP)+   ;;STRIP-OFF ASCII
10608 061472 005766 000002                TST   2(SP)       ;;IS THIS THE FIRST CHAR
10609 061476 001403                BEQ   17$         ;;BRANCH IF YES
10610 061500 006316                ASL   (SP)        ;;NO, SHIFT PRESENT
10611 061502 006316                ASL   (SP)        ;; CHAR OVER TO MAKE
10612 061504 006316                ASL   (SP)        ;; ROOM FOR NEW ONE.
10613 061506 005266 000002      17$:  INC   2(SP)       ;;KEEP COUNT OF CHAR

```

TTY INPUT ROUTINE

```

10614 061512 056616 177776          BIS      -2(SP),(SP)      ;;SET IN NEW CHAR
10615 061516 000703                BR        7$              ;;GET THE NEXT ONE
10616 061520 012767 000442 117022 18$:  MOV      #$QUES,MSGADR
10617 061526 104401                TYPE
10618 061530 000714                BR        20$            ;;SIMULATE CONTROL U
10619                .DSABL  LSB
10620
10621                ;;*****
10622                ;;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
10623                ;;*CALL:
10624                ;;*      RDCHR                ;;INPUT A SINGLE CHARACTER FROM THE TTY
10625                ;;*      RETURN HERE          ;;CHARACTER IS ON THE STACK
10626                ;;*                        ;;WITH PARITY BIT STRIPPED OFF
10627                ;;*
10628                ;;
10629 061532 032767 000100 116312 $RDCHR: BIT      #BIT6,UFDMODE  ;;ARE WE IN QUIET XXDP?          ;PA001
10630 061540 001047                BNE      4$              ;;YES, SO JUST RETURN          ;PA001
10631 061542 011646                MOV      (SP),-(SP)      ;;PUSH DOWN THE PC
10632 061544 016666 000004 000002  MOV      4(SP),2(SP)     ;;SAVE THE PS
10633 061552 105737 177560          1$:  TSTB   @#TKS          ;;WAIT FOR
10634 061556 100375                BPL      1$              ;;A CHARACTER
10635 061560 113766 177562 000004  MOVB    @#TKB,4(SP)      ;;READ THE TTY
10636 061566 042766 177600 000004  BIC     #+C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
10637 061574 026627 000004 000023  CMP     4(SP),#23       ;;IS IT A CONTROL-S?
10638 061602 001013                BNE      3$              ;;BRANCH IF NO
10639 061604 105737 177560          2$:  TSTB   @#TKS          ;;WAIT FOR A CHARACTER
10640 061610 100375                BPL      2$              ;;LOOP UNTIL ITS THERE
10641 061612 113746 177562          MOVB    @#TKB,-(SP)     ;;GET CHARACTER
10642 061616 042716 177600          BIC     #+C177,(SP)    ;;MAKE IT 7-BIT ASCII
10643 061622 022627 000021          CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
10644 061626 001366                BNE      2$              ;;IF NOT DISCARD IT
10645 061630 000750                BR       1$              ;;YES, RESUME
10646 061532 026627 000004 000140  3$:  CMP     4(SP),#140     ;;IS IT UPPER CASE?
10647 061640 002407                BLT      4$              ;;BRANCH IF YES
10648 061642 026627 000004 000175  CMP     4(SP),#175     ;;IS IT A SPECIAL CHAR?
10649 061650 003003                BGT      4$              ;;BRANCH IF YES
10650 061652 042766 000040 000004  BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
10651 061660 000002          4$:  RTI
10652
10653                ;;*****
10654                ;;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
10655                ;;*CALL:
10656                ;;*      RDLIN                ;;INPUT A STRING FROM THE TTY
10657                ;;*      RETURN HERE          ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
10658                ;;*                        ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
10659                ;;*
10660 061662 032767 000100 116162 $RDLIN: BIT      #BIT6,UFDMODE  ;;ARE WE IN QUIET XXDP?          ;PA001
10661 061670 001401                BEQ     11$             ;;NO, SO CONTINUE             ;PA001
10662 061672 000002                RTI
10663                ;;YES, SO RETURN          ;PA001
10664 061674 010346          11$:  MOV     R3,-(SP)        ;;SAVE R3
10665 061676 012703 062016          1$:  MOV     #$TTYIN,R3    ;;GET ADDRESS
10666 061702 022703 062026          2$:  CMP     #$TTYIN+8.,R3 ;;BUFFER FULL?
10667 061706 101405                BLOS    4$              ;;BR IF YES
10668 061710 100407                RDCHR
10669 061712 112613                MOVB    (SP)+,(R3)      ;;GO READ ONE CHARACTER FROM THE TTY
10670 061714 122713 000177          10$:  CMPB   #177,(R3)       ;;GET CHARACTER
10670 061714 122713 000177          10$:  CMPB   #177,(R3)       ;;IS IT A RUBOUT

```



TTY INPUT ROUTINE

```

10671 061720 001005          BNE      3$          ;;SKIP IF NOT
10672 061722 016767 116514 116620 4$:  MOV      $QUES,MSGADR
10673 061730 104401          TYPE          ;;TYPE A '?'
10674 061732 000761          BR        1$          ;;CLEAR THE BUFFER AND LOOP
10675 061734 111367 000054 3$:  MOV      (R3),9$      ;;ECHO THE CHARACTER
10676 061740 012767 062014 116602  MOV      #9$,MSGADR
10677 061746 104401          TYPE
10678 061750 122723 000015  CMP      #15,(R3)+    ;;CHECK FOR RETURN
10679 061754 001352          BNE      2$          ;;LOOP IF NOT RETURN
10680 061756 105063 177777  CLR      1(R3)        ;;CLEAR RETURN (THE 15)
10681 061762 012767 000444 116560  MOV      #$LF,MSGADR
10682 061770 104401          TYPE          ;;TYPE A LINE FEED
10683 061772 012603          MOV      (SP)+,R3     ;;RESTORE R3
10684 061774 011646          MOV      (SP)-,(SP)  ;;ADJUST THE STACK AND PUT ADDRESS OF THE
10685 061776 016666 000004 000002  MOV      4(SP),2(SP) ;;FIRST ASCII CHARACTER ON IT
10686 062004 012766 062016 000004  MOV      #$TTYIN,4(SP)
10687 062012 000002          RTI              ;;RETURN
10688 062014          000          9$:  .BYTE    0          ;;STORAGE FOR ASCII CHAR. TO TYPE
10689 062015          000          .BYTE    0          ;;TERMINATOR
10690 062016          $TTYIN: .BLKB   8.      ;;RESERVE 8 BYTES FOR TTY INPUT
10691 062026          136      125      015  $CNTLU: .ASCIZ  /+U/<15> 12>  ;;CONTROL "U"
10692 062031          012      000
10692 062033          136      107      015  $CNTLG: .ASCIZ  /+G/<15><12>  ;;CONTROL "G"
10693 062036          012      000
10693 062040          136      103      015  $CNTLC: .ASCIZ  /+C/<15><12>  ;;CONTROL "C"
10694 062043          012      000
10694 062045          136      132      015  $CNTLZ: .ASCIZ  /+Z/<15><12>  ;;CONTROL "Z"
10695 062050          012      000
10695 062052          015      012      123  $MSWR: .ASCIZ  <15><12>/SWR = /
10696 062055          127      122      040
10696 062060          075      040      000
10696 062063          040      040      116  $MNEW: .ASCIZ  / NEW = /
10696 062066          105      127      040
10696 062071          075      040      000
10697
10698          ;;*****
10699          .SBTTL  APT COMMUNICATIONS ROUTINE
10700          ;;*****
10701 062074 112767 000001 000214 $ATY1: MOV      #1,$FFLG      ;;TO REPORT FATAL ERROR
10702 062102 112767 000001 000204 $ATY3: MOV      #1,$MFLG      ;;TO TYPE A MESSAGE
10703 062110 000403          BR        $ATYC
10704 062112 112767 000001 000176 $ATY4: MOV      #1,$FFLG      ;;TO ONLY REPORT FATAL ERROR
10705 062120          $ATYC:
10706 062120 010046          MOV      R0,-(SP)      ;;PUSH R0 ON STACK
10707 062122 010146          MOV      R1,(SP)      ;;PUSH R1 ON STACK
10708 062124 105767 000164          TST      $MFLG        ;;SHOULD TYPE A MESSAGE?
10709 062130 001437          BEQ      10$          ;;IF NOT: BR
10710 062132 122767 000001 116512  CMP      #APTENV,$ENV   ;;OPERATING UNDER APT?
10711 062140 001031          BNE      3$          ;;IF NOT: BR
10712 062142 132767 000100 116503  BIT      #APTPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
10713 062150 001425          BEQ      3$          ;;IF NOT: BR
10714 062152 017600 000004          MOV      @4(SP),R0     ;;GET MESSAGE ADDR.
10715 062156 062766 000002 000004  ADD      #2,4(SP)      ;;BUMP RETURN ADDR.
10716 062164 005767 116442 1$:  TST      $MSGTYPE     ;;SEE IF DONE W/ LAST XMISSION?
10717 062170 001375          BNE      1$          ;;IF NOT: WAIT
10718 062172 010067 116450          MOV      R0,$MSGAD    ;;PUT ADDR IN MAILBOX
10719 062176 105720          2$:  TST      (R0)+      ;;FIND END OF MESSAGE

```

APT COMMUNICATIONS ROUTINE

```

10720 062200 001376          BNE      2$
10721 062202 166700 116440   SUB      $MSGAD,RO      ;;SUB START OF MESSAGE
10722 062206 006200          ASR      RO             ;;GET MESSAGE LNTH IN WORDS
10723 062210 010067 116434   MOV      RO,$MSGLGT    ;;PUT LENGTH IN MAILBOX
10724 062214 012767 000004 116410   MOV      #4,$MSGTYPE  ;;TELL APT TO TAKE MSG.
10725 062222 000402          BR       10$
10726 062224 004767 176164   3$:     JSR      PC,$TYPE ;;CALL TYPE MACRO
10727 062230 105767 000062   10$:    TSTB     $FFLG     ;;SHOULD REPORT FATAL ERROR?
10728 062234 001416          BEQ     12$           ;;IF NOT: BR
10729 062236 005767 116410   TST     $ENV         ;;RUNNING UNDER APT?
10730 062242 001413          BEQ     12$           ;;IF NOT: BR
10731 062244 005767 116362   11$:    TST     $MSGTYPE ;;FINISHED LAST MESSAGE?
10732 062250 001375          BNE     11$           ;;IF NOT: WAIT
10733 062252 017667 000004 116354   MOV     @4(SP),$FATAL ;;GET ERROR #
10734 062260 062766 000002 00G004   ADD     #2,4(SP)      ;;BUMP RETURN ADDR.
10735 062266 005267 116340   INC     $MSGTYPE     ;;TELL APT TO TAKE ERROR
10736 062272 105067 000020   12$:    CLRB     $FFLG     ;;CLEAR FATAL FLAG
10737 062276 105067 000013   CLRB     $LFLG     ;;CLEAR LOG FLAG
10738 062302 105067 000006   CLRB     $MFLG     ;;CLEAR MESSAGE FLAG
10739 062306 012601          MOV     (SP)+,R1     ;;POP STACK INTO R1
10740 062310 012600          MOV     (SP)+,RO     ;;POP STACK INTO RO
10741 062312 000207          RTS      PC          ;;RETURN
10742 062314          000          $MFLG: .BYTE 0      ;;MESSG. FLAG
10743 062315          000          $LFLG: .BYTE 0      ;;LOG FLAG
10744 062316          000          $FFLG: .BYTE 0      ;;FATAL FLAG
10745
10746
10747          000200      APTSIZE=200
10748          000001      APTENV=001
10749          000100      APTSPOOL=100
10750          000040      APTCSUP=040
10751
10752          ;;*****
10753          .SBTTL TRAP DECODER
10754          ;;*****
10755          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
10756          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
10757          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
10758          ;*GO TO THAT ROUTINE.
10759
10760 062320 010046          $TRAP: MOV     RO,-(SP) ;;SAVE RO
10761 062322 016600 000002   MOV     2(SP),RO     ;;GET TRAP ADDRESS
10762 062326 005740          TST     -(RO)        ;;BACKUP BY 2
10763 062330 111000          MOVB    (RO),RO     ;;GET RIGHT BYTE OF TRAP
10764 062332 006300          ASL     RO           ;;POSITION FOR INDEXING
10765 062334 016000 062354   MOV     $TRPAD(RO),RO ;;INDEX TO TABLE
10766 062340 000200          RTS      RO         ;;GO TO ROUTINE
10767
10768
10769          ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
10770
10771 062342 011646          $TRAP2: MOV     (SP),-(SP) ;;MOVE THE PC DOWN
10772 062344 016666 000004 000002   MOV     4(SP),2(SP) ;;MOVE THE PSW DOWN
10773 062352 000002          RTI                ;;RESTORE THE PSW
10774
10775          .SBTTL TRAP TABLE
10776

```

TRAP TABLE

```

10777
10778
10779
10780
10781
10782 062354 062342
10783 062356 060414
10784 062360 060724
10785 062362 060700
10786 062364 060740
10787 062366 061274
10788 062370 061144
10789 062372 061532
10790 062374 061662
10791
10792
10793
10794
10795 062376      200      103      112
      062401      113      104      110
      062404      102      040      113
      062407      105      106      061
      062412      061      055      102
      062415      040      103      111
      062420      123      040      104
      062423      111      101      107
      062426      116      117      123
      062431      124      111      103
      062434      200      000
10796 062436      200      101      116
      062441      040      125      116
      062444      105      130      120
      062447      105      103      124
      062452      105      104      040
      062455      124      111      115
      062460      105      055      117
      062463      125      124      040
      062466      124      122      101
      062471      120      040      117
      062474      103      103      125
      062477      122      122      105
      062502      104      000
10797 062504      200      101      116
      062507      040      125      116
      062512      105      130      120
      062515      105      103      124
      062520      105      104      040
      062523      111      114      114
      062526      105      107      101
      062531      114      040      046
      062534      040      122      105
      062537      123      105      122
      062542      126      105      104
      062545      040      111      116
      062550      123      124      122
      062553      125      103      124
      062556      111      117      116

```

```

;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
;*BY THE "TRAP" INSTRUCTION.

```

```

; ROUTINE
;

```

```

$TRPAD: .WORD $TRAP2
        $TYPE  ;;CALL=TYPE   TRAP+1(104401) TTY TYPEOUT ROUTINE
        $TYPOC ;;CALL=TYPOC  TRAP+2(104402) TYPE OCTAL NUMBER (WITH LEADING ZEROS)
        $TYPOS  ;;CALL=TYPOS  TRAP+3(104403) TYPE OCTAL NUMBER (NO LEADING ZEROS)
        $TYPON  ;;CALL=TYPON  TRAP+4(104404) TYPE OCTAL NUMBER (AS PER LAST CALL)
        $GTSWR  ;;CALL=GTSWR  TRAP+5(104405) GET SOFT-SWR SETTING
        $CKSWR  ;;CALL=CKSWR  TRAP+6(104406) TEST FOR CHANGE IN SOFT-SWR
        $RDCHR  ;;CALL=RDCHR  TRAP+7(104407) TTY TYPEIN CHARACTER ROUTINE
        $RDLIN  ;;CALL=RDLIN  TRAP+10(104410) TTY TYPEIN STRING ROUTINE

```

```

;*****

```

```

.SBTTL TELETYPE MESSAGES

```

```

;*****

```

```

TITLE: .ASCIZ <200>/CJKDHB KEF11-B CIS DIAGNOSTIC/<200>

```

```

TIMMSG: .ASCIZ <200>/AN UNEXPECTED TIME OUT TRAP OCCURRED/

```

```

ILLMSG: .ASCIZ <200>/AN UNEXPECTED ILLEGAL & RESERVED INSTRUCTION TRAP OCCURRED/

```

TELETYPE MESSAGES

	062561	040	124	122	
	062564	101	120	040	
	062567	117	103	103	
	062572	125	122	122	
	062575	105	104	000	
10798	062600	200	120	117	PWRMSG: .ASCIZ <200>/POWER FAILURE OCCURRED/
	062603	127	105	122	
	062606	040	106	101	
	062611	111	114	125	
	062614	122	105	040	
	062617	117	103	103	
	062622	125	122	122	
	062625	105	104	000	
10799	062630	040	000		SPACE1: .ASCIZ / /
10800	062632	040	040	040	CTL4: .ASCIZ / 23-*04/
	062635	040	040	040	
	062640	062	063	055	
	062643	052	060	064	
	062646	000			
10801	062647	040	040	040	CTL5: .ASCIZ / 23-*05/
	062652	040	040	040	
	062655	062	063	055	
	062660	052	060	065	
	062663	000			
10802	062664	040	040	040	CTL6: .ASCIZ / 23-*06/
	062667	040	040	040	
	062672	062	063	055	
	062675	052	060	066	
	062700	000			
10803	062701	040	040	040	CTL7: .ASCIZ / 23-*07/
	062704	040	040	040	
	062707	062	063	055	
	062712	052	060	067	
	062715	000			
10804	062716	040	040	040	CTL8: .ASCIZ / 23-*08/
	062721	040	040	040	
	062724	062	063	055	
	062727	052	060	070	
	062732	000			
10805	062733	040	040	040	CTL9: .ASCIZ / 23-*09/
	062736	040	040	040	
	062741	062	063	055	
	062744	052	060	071	
	062747	000			
10806					
10807					
10808					*****
10809					.SBTTL TEST ERROR MESSAGES
10810	062750	200	200	105	*****
	062753	122	122	117	ERRHDR: .ASCIZ <200><200>/ERROR PC CTL CHIP # /
	062756	122	040	120	
	062761	103	040	040	
	062764	040	040	103	
	062767	124	114	040	
	062772	103	110	111	
	062775	120	040	043	
	063000	040	040	040	

TEST ERROR MESSAGES

	063003	040	040	040	
	063006	040	040	000	
10811	063011	103	111	123	ERR1: .ASCIZ /CIS INSTRUCTION FAILURE/<200>
	063014	040	111	116	
	063017	123	124	122	
	063022	125	103	124	
	063025	111	117	116	
	063030	040	106	101	
	063033	111	114	125	
	063036	122	105	200	
	063041	000			
10812	063042	103	111	123	ERR2: .ASCIZ /CIS INTERRUPT FAILURE/<200>
	063045	040	111	116	
	063050	124	105	122	
	063053	122	125	120	
	063056	124	040	106	
	063061	101	111	114	
	063064	125	122	105	
	063067	200	000		
10813	063071	116	117	040	ERR3: .ASCIZ /NO TRAP ON ILLEGAL CIS CLASS OPCODE/<200>
	063074	124	122	101	
	063077	120	040	117	
	063102	116	040	111	
	063105	114	114	105	
	063110	107	101	114	
	063113	040	103	111	
	063116	123	040	103	
	063121	114	101	123	
	063124	123	040	117	
	063127	120	103	117	
	063132	104	105	200	
	063135	000			
10814					
10815					*****
10816					.SBTTL MEMORY MANAGEMENT MESSAGES
10817					*****
10818					
10819	063136	115	105	115	MMDIS: .ASCIZ/MEMORY MANAGEMENT WAS DISABLED/<200>
	063141	117	122	131	
	063144	040	115	101	
	063147	116	101	107	
	063152	105	115	105	
	063155	116	124	040	
	063160	127	101	123	
	063163	040	104	111	
	063166	123	101	102	
	063171	114	105	104	
	063174	200	000		
10820	063176	115	105	115	MMENA: .ASCIZ/MEMORY MANAGEMENT WAS ENABLED/<200>
	063201	117	122	131	
	063204	040	115	101	
	063207	116	101	107	
	063212	105	115	105	
	063215	116	124	040	
	063220	127	101	123	
	063223	040	105	116	
	063226	101	102	114	

MEMORY MANAGEMENT MESSAGES

```

063231      105      104      200
063234      000

10821
10822      .EVEN
10823      ;*****
10824      .SBTTL  BUFFERS & TABLES
10825      ;*****
10826      ;
10827 063236  BUFF1:  .BLKB   100      ;CHARACTER STRING BUFFER 1
10828 063336  BUFF2:  .BLKB   100      ;CHARACTER STRING BUFFER 2
10829 063436  TRANS:  .BLKB   400      ;TRANSLATION TABLE
10830 064036  CHRSET: .BLKB   400      ;CHARACTER SET TABLE
10831
10832
10833      .ODD
10834 064437      000      .BYTE   0      ;RESERVED FOR SIGN FOR LEADING SEPARATE STRINGS
10835 064440  DSBUF1: .BLKB   40      ;DECIMAL STRING BUFFER 1
10836
10837      .ODD
10838 064501      000      .BYTE   0      ;RESERVED FOR SIGN FOR LEADING SEPARATE STRINGS
10839 064502  DSBUF2: .BLKB   40      ;DECIMAL STRING BUFFER 2
10840
10841      .ODD
10842 064543      000      .BYTE   0      ;RESERVED FOR SIGN FOR LEADING SEPARATE STRINGS
10843 064544  DS.DST: .BLKB   40      ;DESTINATION DECIMAL STRING BUFFER
10844
10845      .ODD
10846 064605      000      .BYTE   0      ;RESERVED FOR SIGN FOR LEADING SEPARATE STRINGS
10847 064606  DS.EXP: .BLKB   40      ;DECIMAL STRING BUFFER OF EXPECTED RESULTS
10848
10849
10850      ;*****
10851      .SBTTL  TEST CASES FOR CMPN AND CMPP
10852      ;*****
10853      ;
10854
10855 064646 000007  TCMP:   .WORD   7      ;# OF TEST CASES FOR CMPN & CMPP
10856
10857 064650 000000      .WORD   0      ;SRC1 LENGTH
10858 064652 064740      .WORD  CMP1      ;SRC1 POINTER
10859 064654 000000      .WORD   0      ;SRC2 LENGTH
10860 064656 064744      .WORD  CMP1A     ;SRC2 POINTER
10861
10862 064660 000007      .WORD   7      ;TEST CASE NUMBER 2
10863 064662 064750      .WORD  CMP2
10864 064664 000000      .WORD   0
10865 064666 064762      .WORD  CMP2A
10866
10867 064670 000000      .WORD   0      ;TEST CASE NUMBER 3
10868 064672 064766      .WORD  CMP3
10869 064674 000020      .WORD   20
10870 064676 064772      .WORD  CMP3A
10871
10872 064700 000011      .WORD   11     ;TEST CASE NUMBER 4
10873 064702 065014      .WORD  CMP4
10874 064704 000011      .WORD   11
10875 064706 065030      .WORD  CMP4A

```

TEST CASES FOR CMPN AND CMPP

```

10876
10877 064710 000006 .WORD 6 ;TEST CASE NUMBER 5
10878 064712 065044 .WORD CMP5
10879 064714 000006 .WORD 6
10880 064716 065054 .WORD CMP5A
10881
10882 064720 000001 .WORD 1 ;TEST CASE NUMBER 6
10883 064722 065064 .WORD CMP6
10884 064724 000001 .WORD 1
10885 064726 065070 .WORD CMP6A
10886
10887 064730 000006 .WORD 6 ;TEST CASE NUMBER 7
10888 064732 065074 .WORD CMP7
10889 064734 000005 .WORD 5
10890 064736 065104 .WORD CMP7A
10891
10892
10893 064740 000000 CMP1: .WORD 0 ;SRC1 SIGN
10894 064742 000 .BYTE ;DECIMAL STRING FOR SRC1
10895 .EVEN
10896 064744 000000 CMP1A: .WORD 0 ;SRC2 SIGN
10897 064746 000 .BYTE ;DECIMAL STRING FOR SRC2
10898 .EVEN
10899
10900 064750 000000 CMP2: .WORD 0 ;SRC1 SIGN
10901 064752 007 006 005 .BYTE 007,006,005,004,003,002,001 ;DECIMAL STRING FOR SRC1
10901 064755 004 003 002
10901 064760 001
10902
10903 064762 000000 CMP2A: .EVEN
10904 064764 000 .WORD 0 ;SRC2 SIGN
10905 .BYTE ;DECIMAL STRING FOR SRC2
10906 .EVEN
10907 064766 177777 CMP3: .WORD -1 ;SRC1 SIGN
10908 064770 000 .BYTE ;DECIMAL STRING FOR SRC1
10909 .EVEN
10910 064772 177777 CMP3A: .WORD -1 ;SRC2 SIGN
10911 064774 001 000 011 .BYTE 001,000,011,000,002,000,010,000,003,000,007,000,004,000,004,000
10911 064777 000 002 000
10911 065002 010 000 003
10911 065005 000 007 000
10911 065010 004 000 004
10911 065013 000
;DECIMAL STRING FOR SRC2
10912 .EVEN
10913
10914
10915 065014 177777 CMP4: .WORD -1 ;SRC1 SIGN
10916 065016 001 002 003 .BYTE 001,002,003,000,005,006,007,010,011 ;DECIMAL STRING FOR SRC1
10916 065021 000 005 006
10916 065024 007 010 011
10917 .EVEN
10918 065030 177777 CMP4A: .WORD -1 ;SRC2 SIGN
10919 065032 001 002 003 .BYTE 001,002,003,004,005,006,007,010,011 ;DECIMAL STRING FOR SRC2
10919 065035 004 005 006
10919 065040 007 010 011
10920 .EVEN
10921

```

TEST CASES FOR CMPN AND CMPP

10922	065044	000000				CMP5:	.WORD	0		;SRC1 SIGN
10923	065046	002	001	004			.BYTE	002,001,004,011,040,007		;DECIMAL STRING FOR SRC1
	065051	011	040	007						
10924							.EVEN			
10925	065054	000000				CMP5A:	.WORD	0		;SRC2 SIGN
10926	065056	002	001	004			.BYTE	002,001,004,011,040,007		;DECIMAL STRING FOR SRC2
	065061	011	040	007						
10927							.EVEN			
10928										
10929	065064	177777				CMP6:	.WORD	-1		;SRC1 SIGN
10930	065066	004					.BYTE	004		;DECIMAL STRING FOR SRC1
10931							.EVEN			
10932	065070	000000				CMP6A:	.WORD	0		;SRC2 SIGN
10933	065072	004					.BYTE	004		;DECIMAL STRING FOR SRC2
10934							.EVEN			
10935										
10936	065074	000000				CMP7:	.WORD	0		;SRC1 SIGN
10937	065076	002	003	004			.BYTE	002,003,004,005,006,007		;DECIMAL STRING FOR SRC1
	065101	005	006	007						
10938							.EVEN			
10939	065104	000000				CMP7A:	.WORD	0		;SRC2 SIGN
10940	065106	003	004	005			.BYTE	003,004,005,006,007		;DECIMAL STRING FOR SRC2
	065111	006	007							
10941							.EVEN			

```

.SBTTL TEST CASES FOR ASHN AND ASHP
;*****
;
; TABLE OF TEST CASES FOR ASHN AND ASHP
;
;*****

```

10942										
10943										
10944										
10945										
10946										
10947										
10948										
10949										
10950	065114	000017				TASH:	.WORD	15.		;# OF TEST CASES FOR ASHN & ASHP
10951										
10952	065116	000005					.WORD	5		;SRC LENGTH
10953	065120	000005					.WORD	5		;DST LENGTH
10954	065122	000					.BYTE	0		;SHIFT COUNT
10955	065123	000					.BYTE	0		;ROUNDING DIGIT
10956										
10957	065124	000000					.WORD	0		;TEST CASE NUMBER 2
10958	065126	000017					.WORD	17		
10959	065130	000					.BYTE	0		
10960	065131	000					.BYTE	0		
10961										
10962	065132	000032					.WORD	32		;TEST CASE NUMBER 3
10963	065134	000032					.WORD	32		
10964	065136	000					.BYTE	0		
10965	065137	007					.BYTE	7		
10966										
10967	065140	000025					.WORD	25		;TEST CASE NUMBER 4
10968	065142	000025					.WORD	25		
10969	065144	377					.BYTE	-1		
10970	065145	005					.BYTE	5		
10971										
10972	065146	000014					.WORD	14		;TEST CASE NUMBER 5
10973	065150	000014					.WORD	14		
10974	065152	377					.BYTE	-1		



## TEST CASES FOR ASHN AND ASHP

10975	065153	006	.BYTE	6	
10976					
10977	065154	000006	.WORD	6	;TEST CASE NUMBER 6
10978	065156	000006	.WORD	6	
10979	065160	001	.BYTE	1	
10980	065161	006	.BYTE	6	
10981					
10982	065162	000010	.WORD	10	;TEST CASE NUMBER 7
10983	065164	000011	.WORD	11	
10984	065166	001	.BYTE	1	
10985	065167	000	.BYTE	0	
10986					
10987	065170	000005	.WORD	5	;TEST CASE NUMBER 8
10988	065172	000006	.WORD	6	
10989	065174	002	.BYTE	2	
10990	065175	000	.BYTE	0	
10991					
10992	065176	000016	.WORD	16	;TEST CASE NUMBER 9
10993	065200	000004	.WORD	4	
10994	065202	363	.BYTE	15	
10995	065203	000	.BYTE	0	
10996					
10997	065204	000020	.WORD	20	;TEST CASE NUMBER 10
10998	065206	000010	.WORD	10	
10999	065210	360	.BYTE	-20	
11000	065211	000	.BYTE	0	
11001					
11002	065212	000012	.WORD	12	;TEST CASE NUMBER 11
11003	065214	000007	.WORD	7	
11004	065216	006	.BYTE	6	
11005	065217	000	.BYTE	0	
11006					
11007	065220	000037	.WORD	37	;TEST CASE NUMBER 12
11008	065222	000030	.WORD	30	
11009	065224	030	.BYTE	30	
11010	065225	000	.BYTE	0	
11011					
11012	065226	000020	.WORD	20	;TEST CASE NUMBER 13
11013	065230	000010	.WORD	10	
11014	065232	373	.BYTE	-5	
11015	065233	000	.BYTE	0	
11016					
11017	065234	000005	.WORD	5	;TEST CASE NUMBER 14
11018	065236	000005	.WORD	5	
11019	065240	373	.BYTE	5	
11020	065241	006	.BYTE	6	
11021					
11022	065242	000007	.WORD	7	;TEST CASE NUMBER 15
11023	065244	000007	.WORD	7	
11024	065246	366	.BYTE	-12	
11025	065247	006	.BYTE	6	
11026					
11027					

ADDN/SUBN TEST DATA

```

11029 .SBTTI ADDN/SUBN TEST DATA
11030 ;*****
11031 ;
11032 ; TABLES FOR ADDN AND SUBN
11033 ;
11034 ;*****
11035
11036
11037 065250 000006 TADD: .WORD 6 ;# OF TEST CASES
11038
11039 065252 000020 .WORD 16. ;SRC1 LENGTH
11040 065254 065426 .WORD ADD1 ;SRC1 PTR
11041 065256 000021 .WORD 17. ;SRC2 LENGTH
11042 065260 065450 .WORD ADD1A ;SRC2 PTR
11043 065262 000024 .WORD 20. ;DEST LENGTH
11044 065264 065474 .WORD ADD1B ;DEST PTR (SUM)
11045 065266 000000 .WORD 0 ;CC.EXP (SUM)
11046 065270 065522 .WORD ADD1C ;DEST PTR (DIFF)
11047 065272 000000 .WORD 0 ;CC.EXP (DIFF)
11048
11049 065274 000012 .WORD 10. ;SRC1 >SRC2, SUM +,DIFF
11050 065276 065550 .WORD ADD2
11051 065300 000012 .WORD 10.
11052 065302 065564 .WORD ADD2A
11053 065304 000012 .WORD 10.
11054
11055 065306 065600 .WORD ADD2B
11056 065310 000000 .WORD 0
11057 065312 065614 .WORD ADD2C
11058 065314 000010 .WORD 10
11059
11060 065316 000013 .WORD 11. ;SRC1 +,SRC2 , SUM & DIFF
11061 065320 065630 .WORD ADD3
11062 065322 000013 .WORD 11.
11063 065324 065646 .WORD ADD3A
11064 065326 000014 .WORD 12.
11065 065330 065664 .WORD ADD3B
11066 065332 000010 .WORD 10
11067 065334 065702 .WORD ADD3C
11068 065336 000010 .WORD 10
11069
11070 065340 000020 .WORD 16. ;SRC1 ,SRC2 ,SUM , DIFF .
11071 065342 065720 .WORD ADD4
11072 065344 000003 .WORD 3
11073 065346 065742 .WORD ADD4A
11074 065350 000020 .WORD 16.
11075 065352 065750 .WORD ADD4B
11076 065354 000010 .WORD 10
11077 065356 065772 .WORD ADD4C
11078 065360 000000 .WORD 0
11079
11080 065362 000016 .WORD 14. ;SRC1 = +,SRC2 0,DEST TRUNCATED WITH
11081 ;SUM = +,DIFF =
11082 065364 066014 .WORD ADD5
11083 065366 000001 .WORD 1
11084 065370 066034 .WORD ADD5A
11085 065372 000011 .WORD 9.

```

ADDN/SUBN TEST DATA

11086	065374	066040				.WORD	ADD5B	
11087	065376	000002				.WORD	2	
11088	065400	066054				.WORD	ADD5C	
11089	065402	000012				.WORD	12	
11090								
11091	065404	000010				.WORD	8.	;DEST LENGTH = 0
11092	065406	066070				.WORD	ADD6	
11093	065410	000011				.WORD	9.	
11094	065412	066102				.WORD	ADD6A	
11095	065414	000000				.WORD	0	
11096	065416	066116				.WORD	ADD6B	
11097	065420	000006				.WORD	6	
11098	065422	066116				.WORD	ADD6B	
11099	065424	000006				.WORD	6	
11100								
11101	065426	000000				ADD1: .WORD	0	;SIGN1
11102	065430	005	011	007		.BYTE	5.9..7.4.3.8..1.6.6.7.8..1.2.9..0.6	
	065433	004	007	010				
	065436	001	006	006				
	065441	007	010	001				
	065444	002	011	000				
	065447	006						
11103								;SOURCE STRING 1
11104								
11105								
11106	065450	000000				ADD1A: .EVEN		
11107	065452	001	007	004		.WORD	0	;SIGN2
	065455	003	002	001		.BYTE	1.7.4.3.2.1.6.9..5.3.4.7.9..4.1.6.2	
	065460	006	011	005				
	065463	003	004	007				
	065466	011	004	001				
	065471	006	002					
11108								;SOURCE STRING 2
11109								
11110								
11111	065474	000000				ADD1B: .EVEN		
11112	065476	000	000	000		.WORD	0	;SIGN3
	065501	002	003	004		.BYTE	0.0.0.2.3.4.0.6.5.5.1.2.0.2	
	065504	000	006	005				
	065507	005	001	002				
	065512	000	002					
11113	065514	006	000	007		.BYTE	6.0.7.0.6.8.	
	065517	000	006	010				
11114								;DEST STRING
11115						.EVEN		
11116								
11117	065522	000000				ADD1C: .WORD	0	
11118	065524	000	000	000		.BYTE	0.0.0.1.1.4.5.7.7.8..7.8..6	
	065527	001	001	004				
	065532	005	007	007				
	065535	010	007	010				
	065540	006						
11119	065541	006	011	010		.BYTE	6.9..8..1.2.5.6	
	065544	001	002	005				
	065547	006						
11120						.EVEN		
11121								

## ADDN/SUBN TEST DATA

11122	065550	000000				ADD2:	.WORD	0
11123	065552	003	001	005			.BYTE	3,1,5,9.,2,6,5,3,8.,0
	065555	011	002	006				
	065560	005	003	010				
	065563	000						
11124							.EVEN	
11125								
11126	065564	000000				ADD2A:	.WORD	0
11127	065566	001	007	006			.BYTE	1,7,6,3,2,1,4,1,7,2
	065571	003	002	001				
	065574	004	001	007				
	065577	002						
11128							.EVEN	
11129								
11130	065600	000000				ADD2B:	.WORD	0
11131	065602	004	011	002			.BYTE	4,9.,2,2,4,7,9.,5,5,2
	065605	002	004	007				
	065610	011	005	005				
	065613	002						
11132							.EVEN	
11133								
11134	065614	177777				ADD2C:	.WORD	-1
11135	065616	001	003	011			.BYTE	1,3,9.,6,0,5,1,2,0,8.
	065621	006	000	005				
	065624	001	002	000				
	065627	010						
11136							.EVEN	
11137								
11138	065630	000000				ADD3:	.WORD	0
11139	065632	004	002	003			.BYTE	4,2,3,1,1,7,9.,6,7,9.,6
	065635	001	001	007				
	065640	011	006	007				
	065643	011	006					
11140							.EVEN	
11141								
11142	065646	177777				ADD3A:	.WORD	-1
11143	065650	007	011	003			.BYTE	7,9.,3,6,5,2,2,2,7,8.,1
	065653	006	005	002				
	065656	002	002	007				
	065661	010	001					
11144							.EVEN	
11145								
11146	065664	177777				ADD3B:	.WORD	-1
11147	065666	000	003	007			.BYTE	0,3,7,0,5,3,4,2,5,9.,8.,5
	065671	000	005	003				
	065674	004	002	005				
	065677	011	010	005				
11148							.EVEN	
11149								
11150	065702	177777				ADD3C:	.WORD	-1
11151	065704	001	002	001			.BYTE	1,2,1,6,7,7,0,1,9.,5,7,7
	065707	006	007	007				
	065712	000	001	011				
	065715	005	007	007				
11152							.EVEN	
11153								
11154	065720	177777				ADD4:	.WORD	-1

ADDN/SUBN TEST DATA

11155	065722	004	007	002		.BYTE	4,7,2,9.,3,2,7,7,1,6,1,7,9.,2,8.,4
	065725	011	003	002			
	065730	007	007	001			
	065733	006	001	007			
	065736	011	002	010			
	065741	004					
11156						.EVEN	
11157							
11158	065742	177777			ADD4A:	.WORD	-1
11159	065744	011	003	007		.BYTE	9.,3,7
11160						.EVEN	
11161							
11162	065750	177777			ADD4B:	.WORD	1
11163	065752	004	007	002		.BYTE	4,7,2,9.,3,2,7,7,1,6
	065755	011	003	002			
	065760	007	007	001			
	065763	006					
11164	065764	001	010	000		.BYTE	1,8.,0,2,2,1
	065767	002	002	001			
11165						EVEN	
11166							
11167	065772	000000			ADD4C:	.WORD	0
11168	065774	004	007	002		BYTE	4,7,2,9.,3,2,7,7,1,6
	065777	011	003	002			
	066002	007	007	001			
	066005	006					
11169	066006	001	007	010		.BYTE	1,7,8.,3,4,7
	066011	003	004	007			
11170						.EVEN	
11171							
11172	066014	000000			ADD5:	.WORD	0
11173	066016	004	000	001		.BYTE	4,0,1,7,7,7,9.,6,3,2,5,8.,8.,1
	066021	007	007	007			
	066024	011	006	003			
	066027	002	005	010			
	066032	010	001				
11174						.EVEN	
11175							
11176	066034	000000			ADD5A:	.WORD	0
11177	066036	000				BYTE	0
11178						.EVEN	
11179							
11180	066040	000000			ADD5B:	.WORD	0
11181	066042	007	011	006		.BYTE	7,9.,6,3,2,5,8.,8.,1
	066045	003	002	005			
	066050	010	010	001			
11182						.EVEN	
11183							
11184	066054	177777			ADD5C:	.WORD	-1
11185	066056	007	011	006		.BYTE	7,9.,6,3,2,5,8.,8.,1
	066061	003	002	005			
	066064	010	010	001			
11186						.EVEN	
11187							
11188	066070	000000			ADD6:	.WORD	0
11189	066072	001	007	011		.BYTE	1,7,9.,3,3,9.,8.,1
	066075	003	003	011			

ADDN/SUBN TEST DATA

```

11190      066100      010      001                .EVEN
11191
11192 066102 000000      ADD6A: .WORD 0
11193 066104      010      010      003      .BYTE 8.,8.,3,9.,2,7,6,4,9.
      066107      011      002      007
      066112      006      004      011
11194                .EVEN
11195
11196 066116 000000      ADD6B: .WORD 0
11197 066120      00C      .BYTE 0
11198                .EVEN
11199
11200      ;+
11201      ;      INLINE TEST CASE      (ADDNI & SUBNI)
11202      ;
11203
11204 066122 010020      TADDI: .WORD 010020      ;SRC1 DESCRIPTOR
11205 066124 064440      .WORD DSBUF1
11206 066126 010021      .WORD 010021      ;SRC2 DESCRIPTOR
11207 066130 064502      .WORD DSBUF2
11208 066132 010024      .WORD 010024      ;DEST DESCRIPTOR
11209 066134 064544      .WORD DS.DST
11210
11211      ;+
11212      ;      INLINE TEST CASE      (ADDPI & SUBPI)
11213      ;-
11214
11215 066136 070020      TADDPI: .WORD 070020
11216 066140 064440      .WORD DSBUF1
11217 066142 070021      .WORD 070021
11218 066144 064502      .WORD DSBUF2
11219 066146 070024      .WORD 070024
11220 066150 064544      .WORD DS.DST
11221
11222 066152      INLINE <SPACE: .BLKW 2000>
      066152
11223 072152      INLINE <SPACEY: .BLKW 2000>
      072152
11224                .EVEN
11225
11226                .END      START
      001200
SPACE: .BLKW 2000
SPACEY: .BLKW 2000

```

## SYMBOL TABLE

ABORT	060052	BIT07 =	000200	CMP7A	065104	CVTNL2	025005	DIVPI =	076175
ABORTC	060104	BIT08 =	000400	COUNT	000470	CVTNL3	025007	DOIT	037516
ABORTE	060150	BIT09 =	001000	CR =	000015	CVTNL4	025015	DOIT1	034724
ABORTZ	060174	BIT1 =	000002	CRLF =	000200	CVTNL5	025022	DSBUF1	064440
ADDN =	076050	BIT10 =	002000	CTL	000546	CVTNL6	025032	DSBUF2	064502
ADDNI =	076150	BIT11 =	004000	CTLTBL	060400	CVTNL7	025035	DSLEN1	000514
ADDP =	076070	BIT12 =	010000	CTL4	062632	CVTNL8	025043	DSLEN2	000516
ADDP1 =	076170	BIT13 =	020000	CTL5	062647	CVTNL9	025051	DSTADR =	000000
ADD1	065426	BIT14 =	040000	CT' 6	062664	CVTNP =	076055	DSTMP0	000554
ADD1A	065450	BIT15 =	100000	CTL7	062701	CVTNPI =	076155	DSTMP1	000555
ADD1B	065474	BIT2 =	000004	CTL8	062716	CVTNP1	031140	DSTMP2	000556
ADD1C	065522	BIT3 =	000010	CTL9	062733	CVTNP2	031144	DSTMP3	000557
ADD2	065550	BIT4 =	000020	CVTIL	032210	CVTNP3	031150	DSTMP4	000560
ADD2A	065564	BIT5 =	000040	CVTLI	031304	CVTNP4	031154	DSTMP5	000561
ADD2B	065600	BIT6 =	000100	CVTLN =	076057	CVTNP5	031202	DS.DST	064544
ADD2C	065614	BIT7 =	000200	CVTLNI =	076157	CVTNP6	031214	DS.EXP	064606
ADD3	065630	BIT8 =	000400	CVTLP =	076077	CVTNP7	031230	EMTSV	001200
ADD3A	065646	BIT9 =	001000	CVTLPI =	076177	CVTNP8	031252	EMTVEC =	000030
ADD3B	065664	BPTVEC =	000014	CVTNAA	025205	CVTNP9	031262	END1	035174
ADD3C	065702	BUFF1	063236	CVTNBB	025210	CVTPL =	076073	END2	035170
ADD4	065720	BUFF2	063336	CVTNCC	025213	CVTPLI =	076173	END21	037766
ADD4A	065742	BUFLEN =	000100	CVTNDD	025216	CVTPL1	023164	END22	037762
ADD4B	065750	CBIT =	000001	CVTNEE	025221	CVTPL2	023165	END23	037756
ADD4C	065772	CC	000520	CVTNFF	025224	CVTPL3	023170	END3	035164
ADD5	066014	CC.EXP	000522	CVTNGG	025227	CVTPL4	023173	ERR	000552
ADD5A	066034	CHRSET	064036	CVTNHH	025232	CVTPL5	023201	ERRFLG	000456
ADD5B	066040	CIS4EN	016622	CVTNII	025235	CVTPL6	023205	ERRHDR	062750
ADD5C	066054	CIS5EN	032622	CVTNJJ	025240	CVTPN =	076054	ERROR	057700
ADD6	066070	CIS6EN	045366	CVTNL =	076053	CVTPNI =	076154	ERRTBL	060372
ADD6A	066102	CIS7EN	052766	CVTNLA	025057	CVTPN1	027272	ERRTMP	000474
ADD6B	066116	CIS8EN	054112	CVTNLB	025072	CVTPN2	027273	ERRVEC =	000004
ADR.A	002332	CIS9EN	056100	CVTNLC	025075	CVTPN3	027274	ERR1	063011
ALPHA	002340	CKSWR =	104406	CVTNLD	025100	CVTPN4	027277	ERR2	063042
ALTBUF	056564	CLRBUF	056456	CVTNLE	025103	CVTPN5	027312	ERR3	063071
ALTBU1	056630	CLRDSB	056674	CVTNLF	025106	CVTPN6	027316	GAMMA	002350
APTC SU =	000040	CLRREG	056440	CVTNLG	025111	CVTPN7	027322	GTSWR -	104405
APTENV =	000001	CMPC =	076044	CVTNLH	025114	CVTPN8	027325	HT	000011
APTSIZ =	000200	CMPCI =	076144	CVTNLI =	076153	CVTPN9	027333	ILLMSG	062504
APTSPO =	000100	CMPN -	076052	CVTNLJ	025122	DATA00	000562	ILLOP =	076033
ASHN =	076056	CMPI =	076152	CVTNLK	025125	DATA01	000564	ILLRES	056406
ASHNI =	076156	CMPI1	042330	CVTNLL	025130	DATA02	000566	ILLTST	016474
ASHP =	076076	CMPP =	076072	CVTNLM	025133	DATA03	000570	INC1	033332
ASHPI =	076176	CMPII =	076172	CVTNLN	025136	DATA04	000572	INC2	033716
ASSEMB =	000010	CMPII1	051162	CVTNLO	025141	DATA05	000574	INC21	036120
ASWREG =	000000	CMP1	064740	CVTNLP	025144	DATA06	000576	INC22	036504
BEGIN	001640	CMP1A	064744	CVTNLQ	025147	DATTYP	000464	INC23	037074
BETA	002344	CMP2	064750	CVTNLR	025152	DELAY	000472	INC3	034302
BFADRS	000542	CMP2A	064762	CVTNLS	025155	DESCR0	000616	INTFLG	000460
BF.EXP	000540	CMP3	064766	CVTNLT	025160	DESCR1	000620	INTPAS	000462
BIT0 =	000001	CMP3A	064772	CVTNLU	025163	DESCR2	000622	INTST	015366
BIT00 =	000001	CMP4	065014	CVTNLV	025166	DESCR3	000624	IOTVEC =	000020
BIT01 =	000002	CMP4A	065030	CVTNLW	025171	DESCR4	000626	KIPAR0 =	172340
BIT02 =	000004	CMP5	065044	CVTNLX	025174	DESCR5	000630	KIPAR1 =	172342
BIT03 =	000010	CMP5A	065054	CVTNLY	025177	DIAGMC =	000000	KIPAR2 =	172344
BIT04 =	000020	CMP6	065064	CVTNLZ	025202	DISPLA	000430	KIPAR3 =	172346
BIT05 =	000040	CMP6A	065070	CVTNL\$	025117	DISPRE	000174	KIPAR4 =	172350
BIT06 =	000100	CMP7	065074	CVTNL1	025003	DIVP =	076075	KIPAR5 =	172352

## SYMBOL TABLE

KIPAR6=	172354	MUL2C	055672	SAVR3	000606	SW9	=	001000	TSCNCI	012424		
KIPAR7=	172356	MUL3	055710	SAVR4	000610	TAB89	=	055262	TSKPC	011436		
KIPDR0=	172300	MUL3A	055720	SAVR5	000612	TADD	=	065250	TSKPCI	011454		
KIPDR1=	172302	MUL3B	055724	SAV30	001266	TADDI	=	066122	TSPANCI	013332		
KIPDR2=	172304	MUL3C	055734	SAV32	001270	TADDPI	=	066136	TSPNCI	013350		
KIPDR3=	172306	MUL4	055744	SCANC	=	076042	TASH	=	065114	TSTCAS	000544	
KIPDR4=	172310	MUL4A	055754	SCANCI=	076142	TASHNI	=	045356	TYPE	=	104401	
KIPDR5=	172312	MUL4B	055762	SCOPE	=	000004	TASHPI	=	052756	TYPOC	=	104402
KIPDR6=	172314	MUL4C	055770	SETBUF	=	056520	TBITVE=	000014	TYPC,	=	104404	
KIPDR7=	172316	MUL5	055776	SGNDGT	=	024406	TCMP	=	064646	YPOS	=	104403
LENGTH=	000004	MUL5A	056002	SIGN0	=	000450	TCMPC	=	014334	UFDLFG	=	001272
LF	=	MUL5B	056012	SIGN1	=	000452	TCMPCI	=	014416	UFDMOD=	000052	
LKS	=	MUL5C	056016	SIGN2	=	000454	TCMPNI	=	042320	UFDSET=	000001	
LOCC	=	MUL7	056022	SKPC	=	076041	TCMPPI	=	051152	UQUIET	=	001274
LOCCI	=	MUL7A	056030	SKPCI	=	076141	TCVLNI	=	022076	VBIT	=	000002
LOOP2	=	MUL7B	056040	SKPTST	=	016430	TCVLPI	=	020336	VMKOR	=	001276
LTCVEC=	000100	MUL7C	056052	SPACE	=	066152	TCVNLI	=	025242	WAIT	=	015742
L200	=	NBIT	=	SPACEY	=	072152	TCVNPI	=	031266	XADDN	=	032712
L201	=	NL1	=	SPACE1	=	062630	TCVPLI	=	023210	XADDP	=	045456
L202	=	NOABRT	=	SPANC	=	076043	TCVPNI	=	027336	XASHN	=	042336
L203	=	NP1	=	SPANCI=	076143	TCVTLN	=	021764	XASHP	=	051166	
L204	=	OFFSET	=	SRCADR=	000002	TCVTLP	=	020224	XCMP	=	013360	
L205	=	OVPNCH	=	SR1	=	177572	TCVTNL	=	024514	XCMPN	=	040262
L206	=	PACK	=	STACK	=	000776	TCVTNP	=	031050	XCMP	=	047566
L207	=	PHALT	=	START	=	001200	TCVTPL	=	023132	XCVTLI	=	032206
L300	=	PLI	=	STKLMT=	177774	TCVTPN	=	027202	XCVTLN	=	020346	
L301	=	PN1	=	SUBN	=	076051	TEMPO	=	000476	XCVTLP	=	016712
L302	=	PRO	=	SUBNI	=	076151	TEMP1	=	000500	XCVTML	=	023230
L303	=	PR1	=	SUBP	=	076071	TEMP2	=	000502	XCVTNP	=	027352
L304	=	PR2	=	SUBPI	=	076171	TEMP3	=	000504	XCVTPL	=	022106
L305	=	PR3	=	SWR	=	000426	TEMP4	=	000506	XCVTPN	=	025264
L306	=	PR4	=	SWREG	=	000176	TEMP5	=	000510	XDIVP	=	054202
L307	=	PR5	=	SW0	=	000001	TIMMSG	=	062436	XLOCC	=	007710
MATC	=	PR6	=	SW00	=	000001	TIMOUT	=	056354	XL2DR	=	001726
MATCI	=	PR7	=	SW01	=	000002	TITLE	=	062376	XL3DR	=	002354
MFPT	=	PS	=	SW02	=	000004	TKB	=	177562	XMATC	=	014426
MMCYL	=	PSW	=	SW03	=	000010	TKS	=	177560	XMOV	=	002754
MMDIS	=	PWRMSG	=	SW04	=	000020	TKVEC	=	000060	XMOVRC	=	004634
MMENA	=	PWRVEC=	000024	SW05	=	000040	TLOCC	=	010544	XMOVTC	=	006250
MMSETU	=	RDCHR	=	SW06	=	000100	TLOCCI	=	010570	XMULP	=	053056
MMVEC	=	RDLIN	=	SW07	=	000200	TMATC	=	015334	XSCANC	=	011460
MOV	=	RESREG	=	SW08	=	000400	TMATCI	=	015356	XSKPC	=	010574
MOVCI	=	RESRTN	=	SW09	=	001000	TMOVC	=	004300	XSPANC	=	012434
MOVRC	=	RESVEC=	000010	SW1	=	000002	TMOVCI	=	004624	XSUBN	=	035470
MOVRCI=	076131	R0.EXP	=	SW10	=	002000	TMOVRC	=	006146	XSUBP	=	046512
MOVTC	=	R1.EXP	=	SW11	=	004000	TMOVTC	=	007606	XXDP	=	000466
MOVTCI=	076132	R2.EXP	=	SW12	=	010000	TMULPI	=	056064	ZBIT	=	000004
MSGADR	=	R3.EXP	=	SW13	=	020000	TMVRCI	=	006240	\$APTHD	=	000300
MULP	=	R4.EXP	=	SW14	=	040000	TMVTCI	=	007700	\$ATYC	=	062120
MULPI	=	R5.EXP	=	SW15	=	100000	TPB	=	177566	\$ATY1	=	062074
MUL1	=	R6	=	SW2	=	000004	TPS	=	177564	\$ATY3	=	062102
MUL1A	=	R7	=	SW3	=	000010	TPSRVC	=	016432	\$ATY4	=	062112
MUL1B	=	SAVPSW	=	SW4	=	000020	TPVEC	=	000064	\$AUTOB	=	000422
MUL1C	=	SAVREG	=	SW5	=	000040	TRANS	=	063436	\$BELL	=	000436
MUL2	=	SAVRO	=	SW6	=	000100	TRAPVE=	000034	\$BGNLE=	177777		
MUL2A	=	SAVR1	=	SW7	=	000200	TRTVEC=	000014	\$CHARC	=	060674	
MUL2B	=	SAVR2	=	SW8	=	000400	TSCANC	=	012406	\$CKSWR	=	061144



SYMBOL TABLE

\$CMTAG	000400	\$ISK0 =	000001	\$NSK6 =	000300	\$\$ERFL =	000000	\$50055	002724
\$CNT_C	062040	\$ISK1 =	000001	\$NSK7 =	000300	\$\$FLAG =	000001	\$50056	002736
\$CNTLG	062033	\$ISK2 =	000001	\$NULL	000432	\$\$FROM =	000000	\$50057	002750
\$CNTLU	062026	\$ISK3 =	000001	\$OCNT	061140	\$\$LOC =	057340	\$50062	003104
\$CNTLZ	062045	\$ISK4 =	000001	\$OMODE	061142	\$\$LOCN =	000000	\$50063	003106
\$CPUOP	000660	\$ISK5 =	000001	\$OVER	057500	\$\$REG =	177777	\$50064	003144
\$CRLF	000443	\$LF	000444	\$PASS	000640	\$\$RETU =	000000	\$50065	003152
\$DEVCT	000642	\$LFLG	062315	\$PASTM	000306	\$\$RTN1 =	052477	\$50066	003162
\$DOAGN	056242	\$LOCTA =	000001	\$PWRAD	057662	\$\$RTN2 =	052500	\$50067	003226
\$ENDAD	056232	\$LPADR	000410	\$PWRDN	057516	\$\$SRC =	000000	\$50070	003250
\$ENDCT	056306	\$LPERR	000412	\$PWRUP	057570	\$\$TGSV =	051546	\$50071	003244
\$ENDMG	056313	\$LSTIN =	000001	\$QUES	000442	\$\$TGS1 =	000001	\$50072	003250
\$ENULL	056310	\$LSTTA =	000001	\$RDCHR	061532	\$\$TGS2 =	000000	\$50073	003270
\$ENV	000652	\$MAIL	000632	\$RDLIN	061662	\$\$TO =	000000	\$50074	003312
\$ENVM	000653	\$MAMS1	000662	\$SAVLE =	177777	\$\$\$TAG =	050000	\$50075	003306
\$EOP	056132	\$MBADR	000302	\$SAVR6	057672	\$OFILL	061141	\$50076	003312
\$EOPCT	056304	\$MFLG	062314	\$SCOPE	057376	\$50000	001706	\$50077	003326
\$ERFLG =	000400	\$MNEW	062063	\$SETUP =	000000	\$50001	001706	\$50100	003332
\$ERMAX	000416	\$MSGAD	000646	\$SSK0 =	052507	\$50004	001746	\$50101	003364
\$ERRPC	000420	\$MSGLG	000650	\$SVLAD	057456	\$50005	001774	\$50102	003376
\$ERRTY	060212	\$MSGTY	000632	\$SVPC =	000300	\$50006	002004	\$50103	003410
\$ERTTL	000414	\$MSWR	062052	\$SWREG	000654	\$50007	002052	\$50104	003422
\$ETABL	000652	\$MTYP1	000663	\$TAGLE =	177777	\$50010	002050	\$50105	003434
\$ETEND	000664	\$MXCNT	057514	\$TAGNU =	052510	\$50011	002066	\$50106	003446
\$FATAL	000634	\$NESTL =	000036	\$TEMP =	000300	\$50012	002066	\$50107	003460
\$FFLG	062316	\$NSK0 =	000300	\$TESTN	000636	\$50013	002100	\$50110	003542
\$FILLC	000434	\$NSK1 =	000300	\$TIMES	000404	\$50014	002112	\$50111	003510
\$FILLS	000433	\$NSK10 =	000300	\$TPFLG	000435	\$50015	002124	\$50112	003540
\$F\$AND =	000310	\$NSK11 =	000300	\$TRAP	062320	\$50016	002136	\$50113	003534
\$F\$BAD =	000401	\$NSK12 =	000300	\$TRAP2	062342	\$50017	002146	\$50114	003644
\$F\$BLA =	000170	\$NSK13 =	000300	\$TRPAD	062354	\$50020	002170	\$50115	003546
\$F\$CAS =	000150	\$NSK14 =	000300	\$TSK0 =	052504	\$50021	002166	\$50116	003576
\$F\$DEC =	000220	\$NSK15 =	000300	\$TSK1 =	052506	\$50022	002202	\$50117	003572
\$F\$DO =	000340	\$NSK16 =	000300	\$TSK10 =	051562	\$50023	002202	\$50120	003616
\$F\$FAL =	000405	\$NSK17 =	000300	\$TSK2 =	052507	\$50024	002250	\$50121	003630
\$F\$G00 =	000400	\$NSK2 =	000300	\$TSK3 =	052322	\$50025	002262	\$50122	003644
\$F\$IF =	000110	\$NSK20 =	000300	\$TSK4 =	052323	\$50026	002274	\$50123	003760
\$F\$INC =	000210	\$NSK21 =	000300	\$TSK5 =	052325	\$50027	002306	\$50124	003700
\$F\$LOO =	000200	\$NSK22 =	000300	\$TSK6 =	051546	\$50030	002316	\$50125	003706
\$F\$NAM =	000160	\$NSK23 =	000300	\$TSK7 =	051566	\$50031	002326	\$50126	003732
\$F\$NO =	000403	\$NSK24 =	000300	\$TSTM	000304	\$50034	002374	\$50127	003744
\$F\$OR =	000320	\$NSK25 =	000300	\$TSTNM	000402	\$50035	002422	\$50130	003760
\$F\$RTI =	000350	\$NSK26 =	000300	\$TTYIN	062016	\$50036	002432	\$50131	004004
\$F\$RTN =	000300	\$NSK27 =	000300	\$TYPE	060414	\$50037	002500	\$50132	004104
\$F\$SEL =	000140	\$NSK3 =	000300	\$TYPEC	060630	\$50040	002476	\$50133	004104
\$F\$THE =	000330	\$NSK30 =	000300	\$TYPEX	060676	\$50041	002514	\$50134	004116
\$F\$TRU =	000404	\$NSK31 =	000300	\$TYPOC	060724	\$50042	002514	\$50135	004130
\$F\$UNT =	000130	\$NSK32 =	000300	\$TYPON	060740	\$50043	002526	\$50136	004142
\$F\$WHI =	000120	\$NSK33 =	000300	\$TYPOS	060700	\$50044	002540	\$50137	004154
\$F\$YES =	000402	\$NSK34 =	000300	\$UNIT	000644	\$50045	002552	\$50140	004166
\$GET42	056222	\$NSK35 =	000300	\$UNITM	000310	\$50046	002564	\$50141	004200
\$GTSWR	061274	\$NSK36 =	000300	\$USWR	000656	\$50047	002576	\$50142	004214
\$HIBTS	000300	\$NSK37 =	000300	\$\$ARGC =	000006	\$50050	002610	\$50143	004230
\$ICNT	000406	\$NSK4 =	000300	\$\$BYTE =	000403	\$50051	002654	\$50144	004250
\$IFLEV =	177777	\$NSK40 =	000110	\$\$CASE =	000404	\$50052	002666	\$50145	004260
\$ILLUP	057666	\$NSK41 =	000120	\$\$DST =	000000	\$50053	002700	\$50146	004274
\$INTAG	000423	\$NSK5 =	000300	\$\$ELOC =	000402	\$50054	002712	\$50151	004764

## SYMBOL TABLE

\$50152	004766	\$50245	006572	\$50335	010466	\$50430	012734	\$50520	015102
\$50153	005024	\$50246	006576	\$50336	010500	\$50431	013000	\$50521	015114
\$50154	005032	\$50247	006620	\$50337	010510	\$50432	013012	\$50522	015140
\$50155	005042	\$50250	006614	\$50340	010520	\$50433	013024	\$50523	015170
\$50156	005110	\$50251	006634	\$50341	010530	\$50434	013036	\$50524	015234
\$50157	005132	\$50252	006712	\$50342	010540	\$50435	013050	\$50525	015246
\$50160	005126	\$50253	006724	\$50345	010676	\$50436	013062	\$50526	015260
\$50161	005132	\$50254	006736	\$50346	010712	\$50437	013074	\$50527	015270
\$50162	005154	\$50255	006750	\$50347	010744	\$5044	013406	\$5053	023256
\$50163	005176	\$50256	006762	\$50350	010774	\$50440	013120	\$50530	015300
\$50164	005172	\$50257	006774	\$50351	011122	\$50441	013160	\$50531	015310
\$50165	005176	\$50260	007006	\$50352	011134	\$50442	013172	\$50532	015320
\$50166	005236	\$50261	007076	\$50353	011146	\$50443	013234	\$50533	015330
\$50167	005250	\$50262	007046	\$50354	011160	\$50444	013254	\$50536	015414
\$50170	005262	\$50263	007074	\$50355	011172	\$50445	013266	\$50537	015414
\$50171	005274	\$50264	007070	\$50356	011204	\$50446	013276	\$5054	025312
\$50172	005306	\$50265	007176	\$50357	011216	\$50447	013306	\$50540	015502
\$50173	005320	\$50266	007102	\$50360	011242	\$5045	014454	\$50541	015664
\$50174	005332	\$50267	007130	\$50361	011332	\$50450	013316	\$50542	015660
\$50175	005422	\$50270	007124	\$50362	011360	\$50451	013326	\$50543	015700
\$50176	005372	\$50271	007150	\$50363	011372	\$50454	013474	\$50544	015724
\$50177	005420	\$50272	007162	\$50364	011402	\$50455	013500	\$50545	015736
\$50200	005414	\$50273	007176	\$50365	011412	\$50456	013560	\$50546	016022
\$50201	005554	\$50274	007264	\$50366	011422	\$50457	013602	\$50547	016040
\$50202	005422	\$50275	007236	\$50367	011432	\$50460	013712	\$5055	027400
\$50203	005434	\$50276	007250	\$50372	011560	\$50461	013736	\$50550	016372
\$50204	005472	\$50277	007264	\$50373	011570	\$50462	013764	\$50551	016070
\$50205	005526	\$5030	003002	\$50374	011622	\$50463	014016	\$50552	016102
\$50206	005554	\$50300	007310	\$50375	011652	\$50464	014030	\$50553	016114
\$50207	005552	\$50301	007414	\$50376	011746	\$50465	014042	\$50554	016126
\$50210	005636	\$50302	007414	\$50377	011760	\$50466	014054	\$50555	016140
\$50211	005574	\$50303	007426	\$5040	007736	\$50467	014066	\$50556	016152
\$50212	005606	\$50304	007440	\$50400	012040	\$50470	014100	\$50557	016164
\$50213	005636	\$50305	007452	\$50401	012052	\$50471	014112	\$5056	042370
\$50214	005662	\$50306	007464	\$50402	012064	\$50472	014136	\$50560	016214
\$50215	005762	\$50307	007476	\$50403	012076	\$50473	014250	\$50561	016242
\$50216	005762	\$5031	004662	\$50404	012110	\$50474	014260	\$50562	016236
\$50217	005774	\$50310	007510	\$50405	012122	\$50475	014270	\$50563	016262
\$50220	006006	\$50311	007524	\$50406	012134	\$50476	014300	\$50564	016274
\$50221	006020	\$50312	007540	\$50407	012160	\$50477	014310	\$50565	016310
\$50222	006032	\$50313	007554	\$5041	010622	\$5050	032764	\$50566	016372
\$50223	006044	\$50314	007566	\$50410	012220	\$50500	014320	\$50567	016344
\$50224	006056	\$50315	007602	\$50411	012232	\$50501	014330	\$5057	020374
\$50225	006072	\$5032	006276	\$50412	012310	\$50504	014526	\$50570	016356
\$50226	006106	\$50320	010012	\$50413	012330	\$50505	014536	\$50571	016372
\$50227	006116	\$50321	010026	\$50414	012342	\$50506	014624	\$50574	016464
\$50230	006126	\$50322	010060	\$50415	012352	\$50507	014576	\$50575	016472
\$50231	006142	\$50323	010110	\$50416	012362	\$5051	035542	\$50602	016602
\$50234	006424	\$50324	010236	\$50417	012372	\$50510	014746	\$50603	016664
\$50235	006432	\$50325	010250	\$5042	011506	\$50511	014722	\$50604	016664
\$50236	006442	\$50326	010262	\$50420	012402	\$50512	014746	\$50607	017056
\$50237	006510	\$50327	010274	\$50423	012534	\$50513	015020	\$50610	017056
\$50240	006532	\$50330	010306	\$50424	012544	\$50514	015032	\$50611	017462
\$50241	006526	\$50331	010320	\$50425	012576	\$50515	015044	\$50612	017130
\$50242	006532	\$50332	010332	\$50426	012626	\$50516	015056	\$50613	017462
\$50243	006554	\$50333	010356	\$50427	012722	\$50517	015070	\$50614	017160
\$50244	006576	\$50334	010446	\$5043	012462	\$5052	040310	\$50615	017224

## SYMBOL TABLE

#50616	017310	#5071	046564	#50774	022714	#51071	025520	#51164	027502
#50617	017402	#50710	021124	#50775	022726	#51072	025526	#51165	027500
#50620	017336	#50711	021130	#50776	022752	#51073	025544	#51166	027514
#50621	017340	#50712	021160	#50777	023016	#51074	025550	#51167	027566
#50622	017402	#50713	021204	#51000	023026	#51075	025616	#51170	027566
#50623	017430	#50714	021252	#51001	023036	#51076	025674	#5117	027562
#50624	017462	#50715	021222	#51002	023046	#51077	025672	#51172	027566
#50625	017446	#50716	021246	#51003	023056	#51100	025674	#51173	027612
#50626	017462	#50717	021252	#51004	023066	#51101	025710	#51174	027612
#50627	017524	#5072	047614	#51005	023076	#51102	025734	#51175	027654
#50630	017530	#50720	021314	#51006	023112	#51103	025750	#51176	027654
#50631	017612	#50721	021320	#51007	023126	#51104	025756	#51177	027650
#50632	017624	#50722	021402	#51012	023324	#51105	025774	#51200	027720
#50633	017636	#50723	021414	#51013	023330	#51106	026032	#51201	027720
#50634	017650	#50724	021426	#51014	023422	#51107	026044	#51202	027712
#50635	017662	#50725	021440	#51015	023460	#51110	026066	#51203	027750
#50636	017674	#50726	021452	#51016	023444	#51111	026116	#51204	027744
#50637	017706	#50727	021464	#51017	023770	#51112	026170	#51205	027750
#50640	017722	#5073	022134	#51020	023520	#51113	026170	#51206	030004
#50641	017750	#50730	021476	#51021	023560	#51114	026164	#51207	027776
#50642	017746	#50731	021512	#51022	023560	#51115	026170	#51210	030004
#50643	017774	#50732	021540	#51023	023600	#51116	026230	#51211	030104
#50644	020040	#50733	021536	#51024	023710	#51117	026222	#51212	030134
#50645	020050	#50734	021564	#51025	023614	#51120	026356	#51213	030122
#50646	020060	#50735	021630	#51026	023630	#51121	026264	#51214	030130
#50647	020070	#50736	021640	#51027	023640	#51122	026356	#51215	030134
#50650	020100	#50737	021650	#51030	023702	#51123	026316	#51216	030150
#50651	020110	#5074	053130	#51031	023726	#51124	026352	#51217	030212
#50652	020120	#50740	021660	#51032	023746	#51125	026426	#51220	030212
#50653	020136	#50741	021670	#51033	023770	#51126	026426	#51221	030226
#50654	020136	#50742	021700	#51034	023770	#51127	026420	#51222	030234
#50655	020150	#50743	021710	#51035	024000	#51130	026470	#51223	030252
#50656	020170	#50744	021726	#51036	023776	#51131	026464	#51224	030270
#50657	020174	#50745	021732	#51037	024014	#51132	026470	#51225	030274
#50660	020204	#50746	021744	#51040	024060	#51133	026540	#51226	030336
#50661	020220	#50747	021760	#51041	024052	#51134	026532	#51227	030336
#50664	020624	#5075	054254	#51042	024060	#51135	026540	#51230	030400
#50665	020520	#50752	022202	#51043	024126	#51136	026622	#51231	030376
#50666	020562	#50753	022206	#51044	024140	#51137	026634	#51232	030512
#50667	020546	#50754	022300	#51045	024152	#51140	026646	#51233	030524
#50670	020562	#50755	022564	#51046	024164	#51141	026660	#51234	030536
#50671	020624	#50756	022342	#51047	024176	#51142	026672	#51235	030550
#50672	020610	#50757	022342	#51050	024210	#51143	026704	#51236	030562
#50673	020624	#5076	051220	#51051	024222	#51144	026716	#51237	030574
#50674	020672	#50760	022412	#51052	024246	#51145	026732	#51240	030606
#50675	020650	#50761	022564	#51053	024312	#51146	026760	#51241	030622
#50676	020664	#50762	022512	#51054	024326	#51147	026756	#51242	030650
#50677	021252	#50763	022564	#51055	024332	#51150	027010	#51243	030646
#5070	045530	#50764	022542	#51056	024346	#51151	027056	#51244	030674
#50700	020764	#50765	022556	#51057	024352	#51152	027072	#51245	030740
#50701	020752	#50766	022564	#51060	024366	#51153	027076	#51246	030754
#50702	020764	#50767	022632	#51061	024402	#51154	027112	#51247	030760
#50703	021012	#5077	016740	#51064	025372	#51155	027116	#51250	030774
#50704	021012	#50770	022644	#51065	025376	#51156	027132	#51251	03100
#50705	021070	#50771	022656	#51066	025476	#51157	027146	#51252	0310
#50706	021110	#50772	022670	#51067	025504	#51160	027162	#51253	0310
#50707	021164	#50773	022702	#51070	025526	#51161	027176	#51254	031044

## SYMBOL TABLE

#51255	032206	#51346	032664	#51441	034570	#51534	036630	#51627	040572
#51256	032206	#51347	032664	#51442	034636	#51535	036634	#51630	041040
#51257	031332	#51352	032750	#51443	034642	#51536	036700	#51631	040636
#51260	031460	#51353	032744	#51444	034706	#51537	036744	#51632	040634
#51261	031372	#51354	032764	#51445	034714	#51540	037012	#51633	041040
#51262	031460	#51355	033064	#51446	035022	#51541	037016	#51634	040672
#51263	031502	#51356	033070	#51447	035034	#51542	037062	#51635	040720
#51264	031502	#51357	033124	#51450	035046	#51543	037070	#51636	040720
#51265	031506	#51360	033130	#51451	035060	#51544	037114	#51637	041040
#51266	031532	#51361	033170	#51452	035072	#51545	037156	#51640	040744
#51267	031556	#51362	033174	#51453	035104	#51546	037506	#51641	040760
#51270	031560	#51363	033352	#51454	035116	#51547	037436	#51642	041032
#51271	031604	#51364	033414	#51455	035132	#51550	037372	#51643	041006
#51272	031630	#51365	033712	#51456	035160	#51551	037314	#51644	041030
#51273	031632	#51366	033642	#51457	035156	#51552	037236	#51645	041030
#51274	031656	#51367	033576	#51460	035220	#51553	037170	#51646	041040
#51275	031702	#51370	033532	#51461	035366	#51554	037234	#51647	041056
#51276	031704	#51371	033466	#51462	035402	#51555	037222	#51650	041144
#51277	031730	#51372	033426	#51463	035406	#51556	037230	#51651	041132
#51300	031754	#51373	033452	#51464	035422	#51557	037312	#51652	041136
#51301	031756	#51374	033460	#51465	035426	#51560	037304	#51653	041216
#51302	032000	#51375	033522	#51466	035442	#51561	037370	#51654	041216
#51303	032024	#51376	033566	#51467	035470	#51562	037362	#51655	041204
#51304	032026	#51377	033634	#51470	035466	#51563	037430	#51656	041210
#51305	032046	#51400	033640	#51473	035526	#51564	037434	#51657	041264
#51306	032046	#51401	033704	#51474	035522	#51565	037500	#51660	041264
#51307	032064	#51402	033712	#51475	035542	#51566	037506	#51661	041252
#51310	032066	#51403	033736	#51476	035652	#51567	037614	#51662	041256
#51311	032104	#51404	034000	#51477	035656	#51570	037626	#51663	041324
#51312	032106	#51405	034276	#51500	035712	#51571	037640	#51664	041320
#51313	032124	#51406	034226	#51501	035716	#51572	037652	#51665	041324
#51314	032126	#51407	034162	#51502	035756	#51573	037664	#51666	041372
#51315	032144	#51410	034116	#51503	035762	#51574	037676	#51667	041354
#51316	032154	#51411	034052	#51504	036140	#51575	037710	#51670	041360
#51317	032164	#51412	034012	#51505	036202	#51576	037724	#51671	041414
#51320	032620	#51413	034036	#51506	036500	#51577	037752	#51672	041502
#51321	032620	#51414	034044	#51507	036430	#51600	037750	#51673	041470
#51322	032230	#51415	034106	#51510	036364	#51601	040012	#51674	041474
#51323	032260	#51416	034152	#51511	036320	#51602	040160	#51675	041546
#51324	032262	#51417	034220	#51512	036254	#51603	040174	#51676	041534
#51325	032312	#51420	034224	#51513	036214	#51604	040200	#51677	041540
#51326	032314	#51421	034270	#51514	036240	#51605	040214	#51700	041606
#51327	032344	#51422	034276	#51515	036246	#51606	040220	#51701	041574
#51330	032346	#51423	034322	#51516	036310	#51607	040234	#51702	041600
#51331	032376	#51424	034364	#51517	036354	#51610	040262	#51703	041646
#51332	032400	#51425	034714	#51520	036422	#51611	040260	#51704	041642
#51333	032426	#51426	034644	#51521	036426	#51614	040404	#51705	041646
#51334	032430	#51427	034600	#51522	036472	#51615	040402	#51706	041714
#51335	032452	#51430	034522	#51523	036500	#51616	040416	#51707	041676
#51336	032454	#51431	034444	#51524	036524	#51617	040430	#51710	041702
#51337	032476	#51432	034376	#51525	036572	#51620	040460	#51711	042002
#51340	032500	#51433	034442	#51526	037070	#51621	040456	#51712	042012
#51341	032522	#51434	034430	#51527	037020	#51622	040472	#51713	042022
#51342	032524	#51435	034436	#51530	036754	#51623	040522	#51714	042032
#51343	032546	#51436	034520	#51531	036710	#51624	040572	#51715	042042
#51344	032614	#51437	034512	#51532	036644	#51625	040546	#51716	042052
#51345	032614	#51440	034576	#51533	036604	#51626	040572	#51717	042062

## SYMBOL TABLE

#51720	042104	#52012	044020	#52102	045316	#52201	050040	#52274	051436
#51721	042126	#52013	043700	#52103	045352	#52202	050064	#52275	051464
#51722	042152	#52014	043650	#52104	045350	#52203	050064	#52276	051472
#51723	042240	#52015	043674	#52105	045430	#52204	050332	#52277	051476
#51724	042262	#52016	043700	#52106	045430	#52205	050130	#52300	051512
#51725	042266	#52017	044020	#52111	045514	#52206	050126	#52301	051520
#51726	042310	#5202	041376	#52112	045510	#52207	050332	#52302	052010
#51727	042314	#52020	043742	#52113	045530	#52210	050164	#52303	051652
#51732	042434	#52021	043746	#52114	046002	#52211	050212	#52304	051620
#51733	042450	#52022	043752	#52115	046014	#52212	050212	#52305	051652
#51734	042472	#52023	044006	#52116	046026	#52213	050332	#52306	051710
#51735	042524	#52024	044176	#52117	046040	#52214	050236	#52307	051710
#51736	042562	#52025	044062	#52120	046052	#52215	050252	#52310	051772
#51737	042546	#52026	044054	#52121	046064	#52216	050324	#52311	051760
#51740	042544	#52027	044062	#52122	046076	#52217	050300	#52312	051754
#51741	042562	#52030	044176	#52123	046112	#52220	050322	#52313	051764
#51742	042576	#52031	044114	#52124	046140	#52221	050322	#52314	052010
#51743	042606	#52032	044120	#52125	046136	#52222	050332	#52315	052232
#51744	043164	#52033	044124	#52126	046164	#52223	050404	#52316	052232
#51745	042770	#52034	044160	#52127	046366	#52224	050376	#52317	052076
#51746	042676	#52035	044370	#52130	046402	#52225	050402	#52320	052124
#51747	042670	#52036	044240	#52131	046406	#52226	050474	#52321	052232
#51750	042676	#52037	044246	#52132	046422	#52227	050416	#52322	052170
#51751	042710	#52040	044300	#52133	046426	#52230	050416	#52323	052232
#51752	042710	#52041	044310	#52134	046464	#52231	050444	#52324	052216
#51753	042752	#52042	044316	#52135	046512	#52232	050466	#52325	052222
#51754	042744	#52043	044354	#52136	046510	#52233	050472	#52326	052322
#51755	042770	#52044	044370	#52141	046550	#52234	050542	#52327	052332
#51756	043164	#52045	044572	#52142	046544	#52235	050534	#52330	052336
#51757	043032	#52046	044456	#52143	046564	#52236	050540	#52331	052346
#51760	043102	#52047	044464	#52144	047046	#52237	050632	#52332	052352
#51761	043064	#52050	044502	#52145	047060	#52240	050554	#52333	052400
#51762	043062	#52051	044512	#52146	047072	#52241	050554	#52334	052412
#51763	043100	#52052	044520	#52147	047104	#52242	050602	#52335	052432
#51764	043164	#52053	044556	#52150	047116	#52243	050624	#52336	052462
#51765	043122	#52054	044572	#52151	047130	#52244	050630	#52337	052456
#51766	043120	#52055	044664	#52152	047142	#52245	050720	#52340	052474
#51767	043136	#52056	044700	#52153	047156	#52246	050734	#52341	052520
#51770	043142	#52057	044704	#52154	047204	#52247	050740	#52342	052532
#51771	043164	#52060	044732	#52155	047202	#52250	050754	#52343	052622
#51772	043202	#52061	044744	#52156	047230	#52251	050760	#52344	052636
#51773	043256	#52062	044764	#52157	047442	#52252	051004	#52345	052642
#51774	043306	#52063	045014	#52160	047456	#52253	051072	#52346	052656
#51775	043336	#52064	045010	#52161	047462	#52254	051114	#52347	052662
#51776	043522	#52065	045026	#52162	047476	#52255	051120	#52350	052700
#51777	043406	#52066	045042	#52163	047502	#52256	051142	#52351	052712
#52000	043402	#52067	045040	#52164	047540	#52257	051146	#52352	052716
#52001	043406	#52070	045054	#52165	047566	#52262	051306	#52353	052752
#52002	043522	#52071	045100	#52166	047564	#52263	051300	#52354	052750
#52003	043454	#52072	045112	#52171	047710	#52264	051304	#52355	053030
#52004	043462	#52073	045222	#52172	047706	#52265	051362	#52356	053030
#52005	043466	#52074	045236	#52173	047720	#52266	051322	#52361	053114
#52006	043522	#52075	045242	#52174	047754	#52267	051322	#52362	053110
#52007	043604	#52076	045256	#52175	047752	#52270	051342	#52363	053130
#5201	041040	#52077	045262	#52176	047764	#52271	051356	#52364	053402
#52010	043562	#52100	045300	#52177	050014	#52272	051362	#52365	053414
#52011	043604	#52101	045312	#52200	050064	#52273	051404	#52366	053426

## SYMBOL TABLE

\$52367	053440	\$52413	054240	\$52434	055166	\$52455	056672	\$52476	057252
\$52370	053452	\$52414	054234	\$52435	055172	\$52456	056672	\$52477	057374
\$52371	053464	\$52415	054254	\$52436	055230	\$52457	056650	\$52500	057374
\$52372	053476	\$52416	054536	\$52437	055256	\$52460	056734	\$52501	057306
\$52373	053512	\$52417	054550	\$52440	055254	\$52461	056734	\$52502	057312
\$52374	053540	\$52420	054562	\$52441	056132	\$52462	056714	\$52503	057322
\$52375	053536	\$52421	054574	\$52442	056454	\$52463	056766	\$52504	057374
\$52376	053564	\$52422	054606	\$52443	056454	\$52464	056766	\$52505	057334
\$52377	053766	\$52423	054620	\$52444	056516	\$52465	057020	\$52506	057334
\$52400	054002	\$52424	054632	\$52445	056516	\$52466	057020	\$52507	057362
\$52401	054006	\$52425	054646	\$52446	056476	\$52467	057252	\$5401	026026
\$52402	054022	\$52426	054674	\$52447	056562	\$52470	057252	\$5402	026052
\$52403	054026	\$52427	054672	\$52450	056562	\$52471	057046	\$5403	026764
\$52404	054064	\$52430	054720	\$52451	056540	\$52472	057062	\$5601	043234
\$52405	054112	\$52431	055132	\$52452	056626	\$52473	057062	\$5602	043242
\$52406	054110	\$52432	055146	\$52453	056626	\$52474	057226	\$5603	045032
\$52407	054154	\$52433	055152	\$52454	056604	\$52475	057244	.\$X	= 000300
\$52410	054154								

. ABS. 076152 000  
000000 001

ERRORS DETECTED: 0

VIRTUAL MEMORY USED: 32913 WORDS ( 129 PAGES)

DYNAMIC MEMORY: 20060 WORDS ( 77 PAGES)

ELAPSED TIME: 01:09:37

CJKDHB,CJKDHB/-SP=ORION/ML,SPMAC/ML,CJKDHB